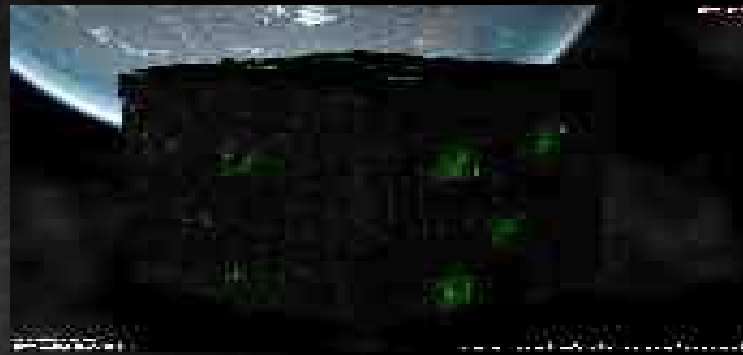


Machine code...in...SPAAACE;
or, So You're Facing an Invasion of
Alien Robots



Resistance is futile

By Asad Sayeed for CMSC212 at the
University of Maryland, College Park.

No tribbles were harmed in the making of this
presentation

Quick! You are being invaded from space by Neptunian Robots



“Destroy!
Destroy!”
(Actually
from Doctor
Who)

- Conveniently they seem to have independently evolved USB connectors! You, the hero, can reprogram them!
- But first you must know what a program really is. Ooops.
- Why do we compile? The computer can't understand your fancy-shmancy C syntax!
- The compiler turns your C code into machine code . . . numerical instructions of fixed width.

Stupid Robots

Duh!
(Or rather)
00000000!

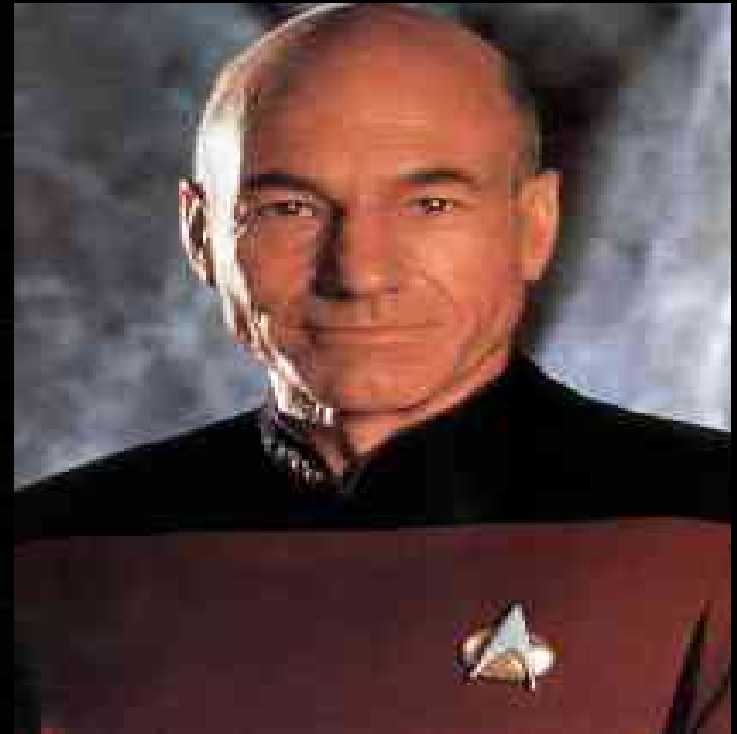
- Our computer doesn't really need much.
 - Set of registers R_n . This is a small working memory for the CPU.
 - Large direct-access sequential memory. Stores programs and data.
 - Small set of instructions to manipulate registers and main memory.
- Instructions stored as a sequence of codes in main memory.



This robot wants
to conquer you,
but it is stupid.
(Actually MST3K)

Make it so

- Instruction:
numerical identifier
plus fixed number of
arguments.
- In assignment, format
is <instruction #,
register#, reg#, reg#,
mem addr>
- Instructions do very
elementary things:
move bytes, add, etc.



He only needs a few
instructions, like “engage”,
and “set a course”, and
“Prime Directive”

Mystery Code!

- Do you really want to write in machine code? No!
- “Assembly language”: human-readable machine code. An “assembler” converts it to machine code.
- Let's see an example.

```
0: Input R2
1: Input R3
2: Move R4 R0
3: Load R5 #1
4: Negate R5
5: Add R2 R5 R6
6: Move R2 R6
7: Add R3 R4 R6
8: Move R4 R6
9: Bnn R2 5
10: Output R4
```



They do not advise that you program in machine code. (www.f9j9.com/iwp2h.htm)

Mystery Code!

- Each assembly language instruction corresponds to a machine code word.

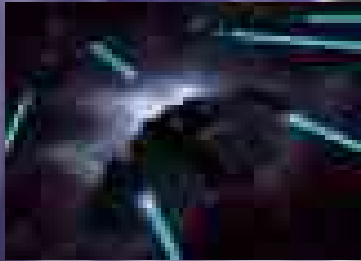
- From assignment, R1 = program counter. Incremented when instruction succeeds. Branch and Bnn can make other changes.

- What does this program do?

0: Input R2
1: Input R3
2: Move R4 R0
3: Load R5 #1
4: Negate R5
5: Add R2 R5 R6
6: Move R2 R6
7: Add R3 R4 R6
8: Move R4 R6
9: Bnn R2 5
10: Output R4

Primitive humans!
You're still just
using punch cards!
([http://valinor.ca/
computing/
img0.htm](http://valinor.ca/computing/img0.htm))

Mystery Code!



Don't worry, you have warded off the attack. You think. (www.b5tech.net/art/bsg/bsgimages.htm)

- This program enables the robots to CONQUER THE UNIVERSE.

- Actually, no, it's just $(a + 1) * b$.
- We used mostly registers and input. What if we wanted to use memory?

```
0: Input R2
1: Input R3
2: Move R4 R0
3: Load R5 #1
4: Negate R5
5: Add R2 R5 R6
6: Move R2 R6
7: Add R3 R4 R6
8: Move R4 R6
9: Bnn R2 5
10: Output R4
```

Oh no! They're evolving!

- We change the Input and Output to Load and Store.
- Note the two forms of Load in the assignment assembly language.
- Note that Bnn is not relative: what if you remove a line?
- What if you remove line (prog addr) 4?

```
0: Load R2 20
1: Load R3 21
2: Move R4 R0
3: Load R5 #1
4: Negate R5
5: Add R2 R5 R6
6: Move R2 R6
7: Add R3 R4 R6
8: Move R4 R6
9: Bnn R2 5
10: Store R4 22
```



Not only are they evolving, they are muuuuultiplying. Machine code will not save you now. (Stargate)

Unfortunately, you have yet to face
the final challenge.

CMSC Submission and Testing Server

WELCOME TO THE UMD COMPUTER SCIENCE SUBMIT AND TESTING SERVER

Login with your Directory ID and password

Directory ID:

Directory Password:

Login

- Your Directory ID is **not** your UMID, SID or SSN.
- Your Directory password is **not** your ARES/Testudo PIN.
- You may discover your Directory ID by visiting:
<https://www.ldap.umd.edu/cgi-bin/chpwd?searchbyumid>
- You may set your Directory password by visiting:
<https://www.ldap.umd.edu/cgi-bin/chpwd>

Will you survive? Are you strong enough?