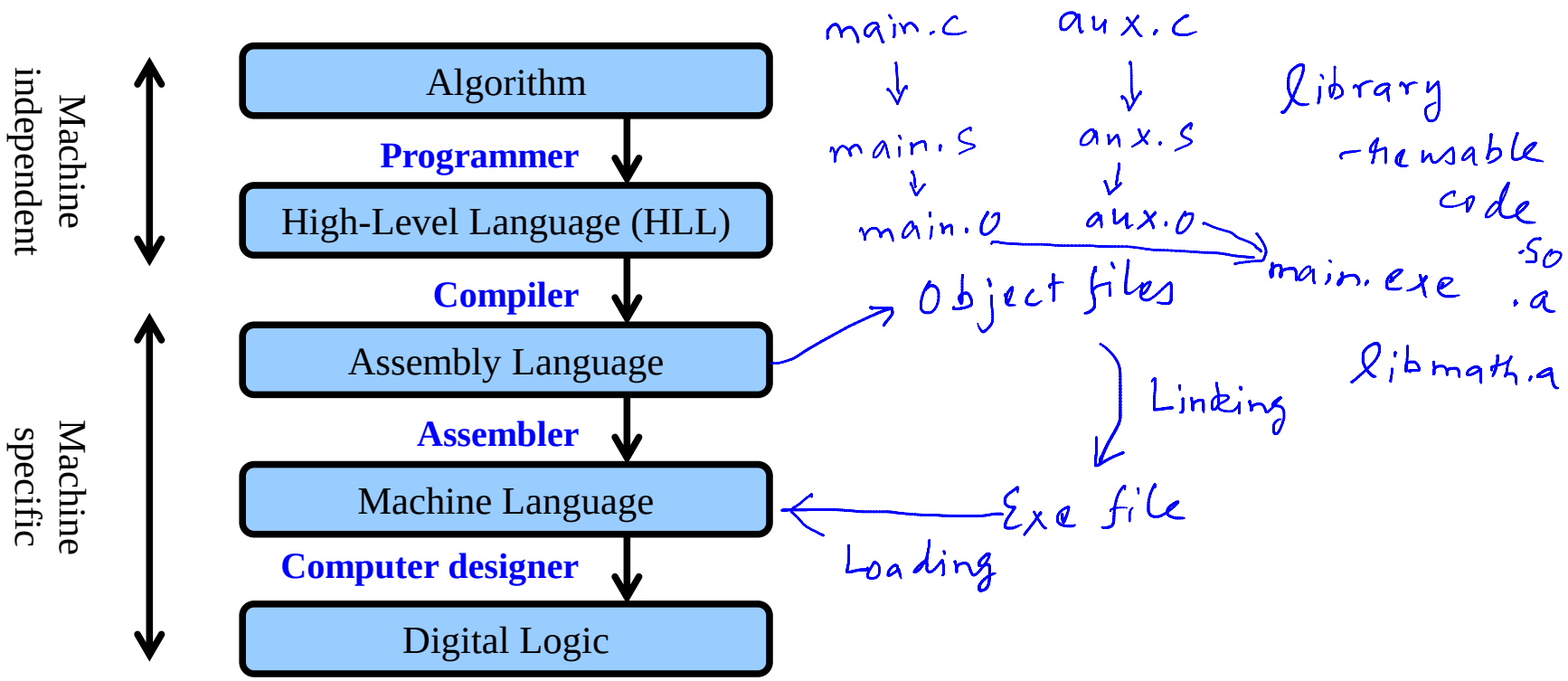
CS305
Computer Architecture

From HLL Code to Process: Object Files, Linking, Loading

Bhaskaran Raman
Room 406, KR Building
Department of CSE, IIT Bombay

http://www.cse.iitb.ac.in/~br

Steps from HLL Code to Running Process



Contents of an Object File

1. Header
2. Text segment
3. Static data
4. Relocation table: list of unresolved instructions
5. Symbol table: table of unresolved symbols
6. Debugging information

Object Files: An Example

main.s

```
.data Y
main:
…
jal   F          # I1
…
lw    $s0, X     # I2
…
G:
…
lw    $s1, Y     # I3
```

aux.s

```
.data X
F:
…
jal   G          # I4
…
lw    $s2, X     # I5
…
…
…
sw    Y, $s3     # I6
```

```
Relocation table: I1,I2,I3
Symbol table: main,F,G,X,Y
```

```
Relocation table: I4,I5,I6
Symbol table: F,G,X,Y
```

Functionalities of Linker, Loader

- Linker: – generate exe file
  - Organize text and data as it would appear in memory
  - Resolve symbols
  - Rewrite instructions referred to in relocation table
- Loader:
  - Place text and data in memory
  - Init stack, other regs, including args
  - Call main

Boot loader
↓
OS
↓
programs

Dynamic Linking

- Link files on demand
- Why?
  - Smaller exe files, better use of memory
  - Newer libraries can be used seamlessly
- How?
  - Rewrite indirect jump address
  - Use jump table of function pointers
  - **Note:** `jalr` instruction needed to support function pointers

Summary

- HLL code → Compiler → Assembler → Linker → Loader → Executing Process
- Object file contents
- Dynamic linking