

# Lecture 8

## Linkers

*CS 241: Foundations of Sequential Programs*  
Fall 2014

Troy Vasiga et al  
University of Waterloo

## Loading Review

Write the MERL file for the following MIPS code:

- A: .word B
- B: .word 7
- C: .word 0xA
- D: .word C

## Loading Review

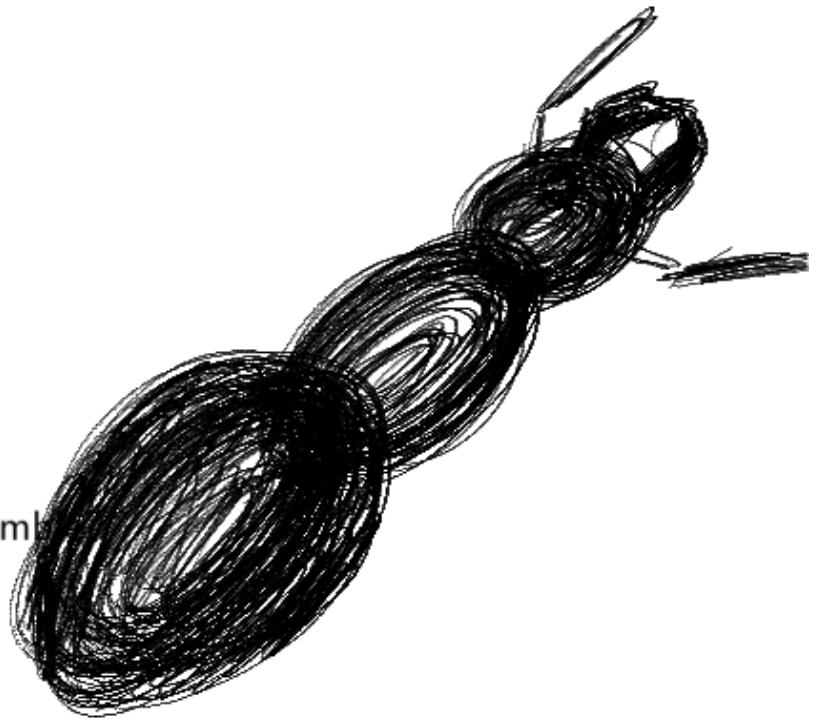
Changes that need to be made to the assembler:

- ▶ Pass 1:

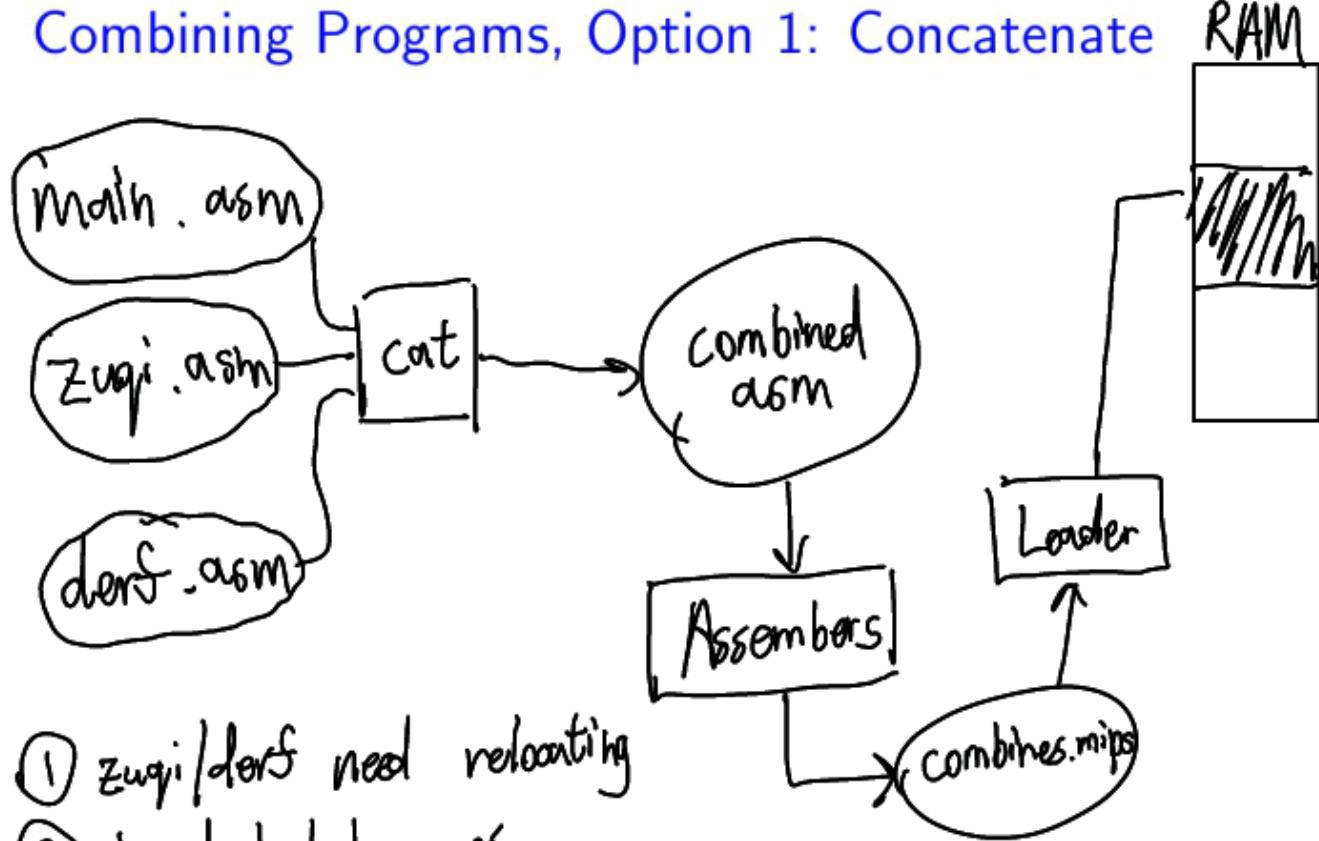
- ▶ Pass 2:

## Combining Programs

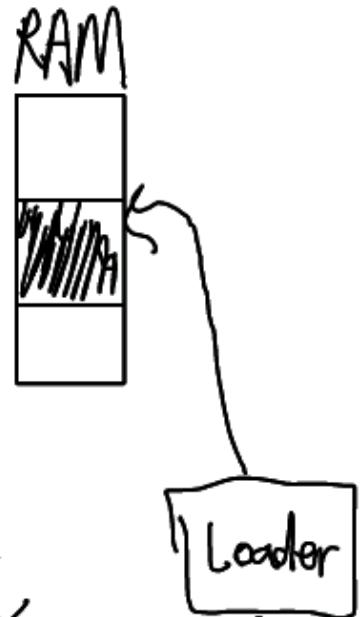
- ▶ Why?
- ▶ When would we want to assemble programs?
- ▶ Example Slides 1 and 2



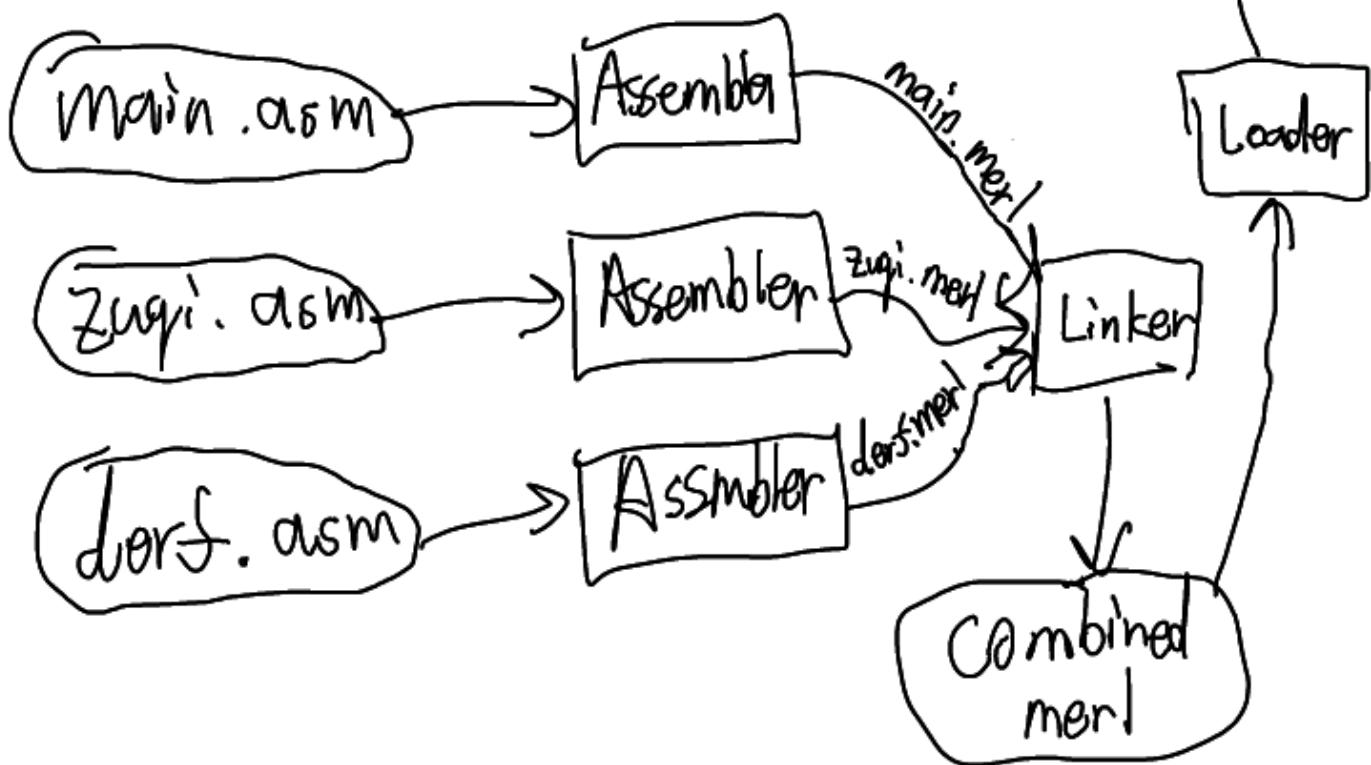
## Combining Programs, Option 1: Concatenate



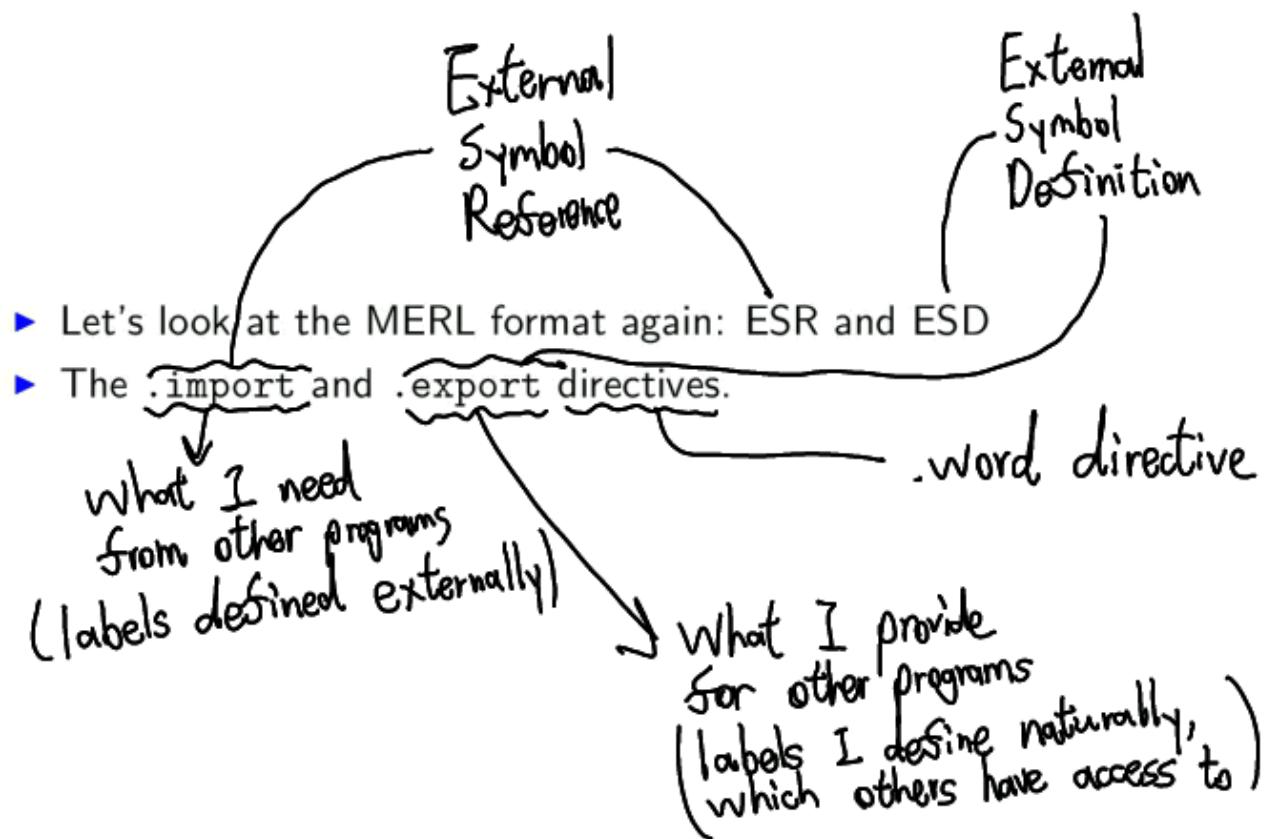
- ① zugi/durf need relocating
- ② shared label names
- ③ errors are hard to track
- ④ Productivity problems



## Combining Programs, Option 2: Linker



## Modifying the assembler



## Pass 1 and Pass 2 modifications

Pass 1 Changes:

- .import: record somewhere (symbol table) each label which is imported.
- .export: record the fact that I will be creating an ESD

Pass 2 Changes:

- .import: write out an ESR record everytime I encounter that external name
- .export: write out an ESD record  
⇒ Look in symbol table.

## MERL Format

Three words

- ▶ Header cookie
- ▶ Length of entire .merl file
- ▶ Length of just MIPS program (and header)
- ▶ Program code
- ▶ Notes of what to change]

See full description on CS241 webpage.

What are these "notes of what to change"?

1. Relocation

.word 1  
.word location ← what to update when loaded  
(with export)

2. ESD (1:1)

.word 5  
.word location ← where defined  
.word n

name { } → n words

3. ESR (1:many with import)

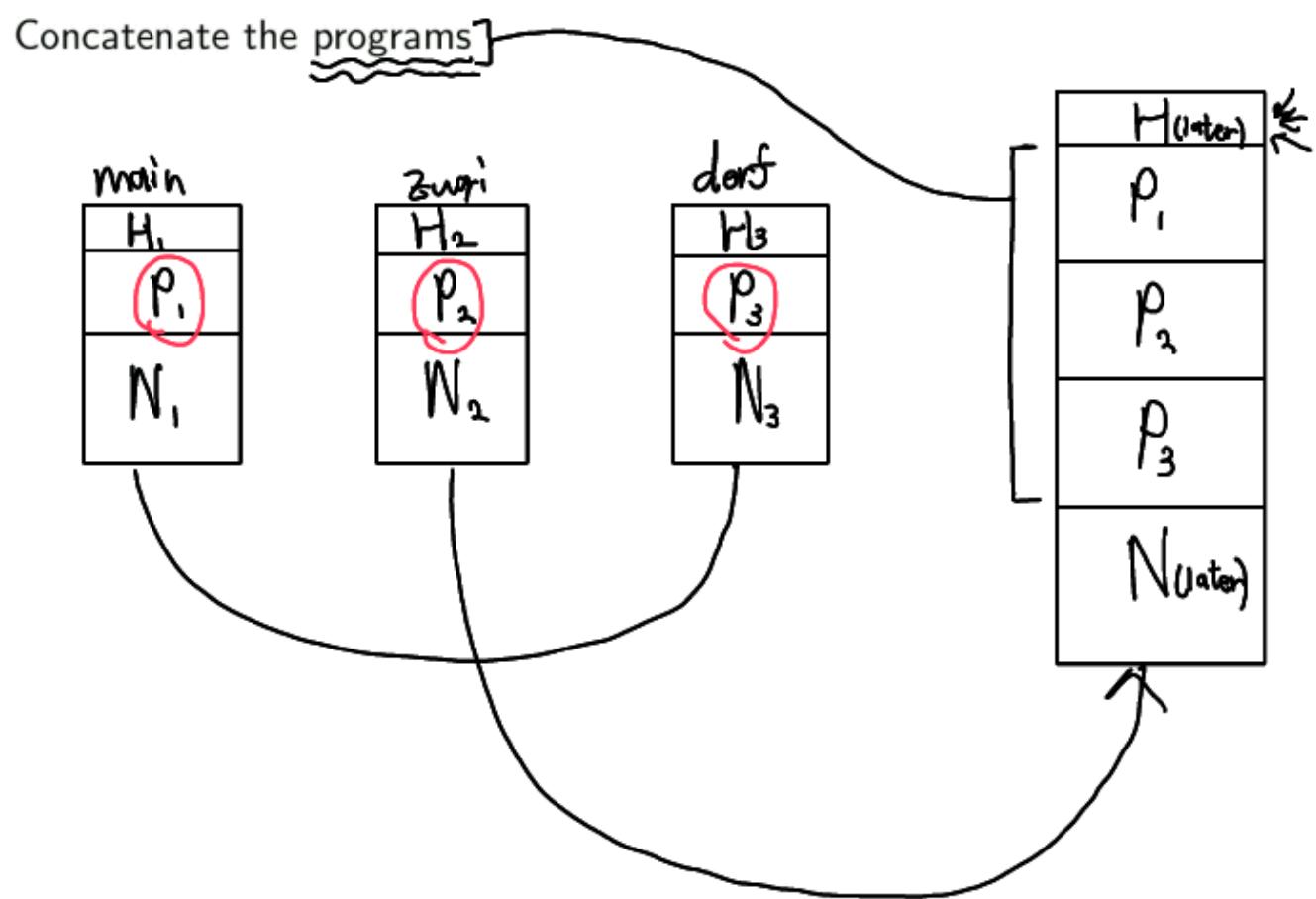
.word 0x11  
.word location ← where used  
.word n

name { } → n words<sup>10</sup>

## Linker Pseudocode

1. Concatenate the programs
2. Construct ESD
3. Use ESR
4. Relocate (internally)

## Linker Pseudocode Step 1



## Linker Pseudocode Step 2

Construct ESD

Combine all the ESDs into one ESD

→ update addresses

main's ( $P_1$ ) ESD  $\Rightarrow$  no change

zmq's ( $P_2$ ) ESD  $\Rightarrow$  add  $|P_1|$  to each address.

dex's ( $P_3$ ) ESD  $\Rightarrow$  add  $|P_1| + |P_2|$  to each address.

## Linker Pseudocode Step 3

Use ESR

For each ESR entry  
    ⇒ look in the new ESD  
        if found     ⇒ update the value at location  
                     plus the offset  
        if not found     ⇒ write out a new ESR  
                     with offsets.

## Linker Pseudocode Step 4

Relocate Internally

why? ex: .word label in A2/A3 is encoded based on a different starting address.

→ each relocation entry:

- ⇒ add the appropriate offset in code.
- ⇒ add the appropriate offset in relocation entry

## Static Linking vs. Dynamic Linking

- ▶ Static linking example:

↳ just seen

↳ all linked before execution

- ▶ Dynamic linking example:

↳ done at execution time

- ① Create a "string" "help"
- ② \$1 to point to this string
- ③ lis \$29
- ④ .word binary Loader → search the libraries  
for what is in \$1  
put the address of  
that function into \$3
- ⑤ jahr \$29
- ⑥ jahr \$3

Windows DLLs

Dynamically Linked Library

Where are we now? Where to next?