Computer Architecture, Fall 2019

C.A.L.L. Example

C.A.L.L. Example

```
#include <stdio.h>
int main(){
    printf("Hello, %s\n","world");
    return 0;
}
```

Compiled Hello.c: Hello.s

```
.text
  .align 2
  .global main
main:
  addi sp,sp,-16
  sw   ra,12(sp)
  lui  a0,%hi(str1)
  addi a0,a0,%lo(str1)
  lui  a1,%hi(str2)
  addi a1,a1,%lo(str2)
  call printf
  lw   ra,12(sp)
  addi sp,sp,16
  li   a0,0
  ret
  .section .rodata
  .balign 4
str1:
  .string "Hello, %s!\n"
str2:
  .string "world"
```

```
# Directive: enter text section
#
#
#
# allocate stack frame
# save return address
# compute address of string1
#
# compute address of string2
#
# call function printf
# restore return address
# deallocate stack frame
# load return value 0
# return
#
#
# label for first string
#
# label for second string
```

Compiled Hello.c: Hello.s

```
.text
  .align 2
  .global main
main:
  addi sp,sp,-16
  sw   ra,12(sp)
  lui  a0,%hi(str1)
  addi a0,a0,%lo(str1)
  lui  a1,%hi(str2)
  addi a1,a1,%lo(str2)
  jal, ra, printf
  lw   ra,12(sp)
  addi sp,sp,16
  li   a0,0
  ret
  .section .rodata
  .balign 4
str1:
  .string "Hello, %s!\n"
str2:
  .string "world"
```

```
# Directive: enter text section
#
#
#
# allocate stack frame
# save return address
# compute address of string1
#
# compute address of string2
#
# call function printf
# restore return address
# deallocate stack frame
# load return value 0
# return
#
#
# label for first string
#
# label for second string
```

Compiled Hello.c: Hello.s

```
.text
  .align 2
  .global main
main:
  addi sp,sp,-16
  sw   ra,12(sp)
  lui  a0,%hi(str1)
  addi a0,a0,%lo(str1)
  lui  a1,%hi(str2)
  addi a1,a1,%lo(str2)
  jal, ra, printf
  lw   ra,12(sp)
  addi sp,sp,16
  addi a0, a0, 0
  ret
  .section .rodata
  .balign 4
str1:
  .string "Hello, %s!\n"
str2:
  .string "world"
```

```
# Directive: enter text section
#
#
#
# allocate stack frame
# save return address
# compute address of string1
#
# compute address of string2
#
# call function printf
# restore return address
# deallocate stack frame
# load return value 0
# return
#
#
# label for first string
#
# label for second string
```

Compiled Hello.c: Hello.s

```
.text
  .align 2
  .global main
main:
  addi sp,sp,-16
  sw   ra,12(sp)
  lui  a0,%hi(str1)
  addi a0,a0,%lo(str1)
  lui  a1,%hi(str2)
  addi a1,a1,%lo(str2)
  jal, ra, printf
  lw   ra,12(sp)
  addi sp,sp,16
  addi a0, a0, 0
  jalr, x0, ra, 0
  .section .rodata
  .balign 4
str1:
  .string "Hello, %s!\n"
str2:
  .string "world"
```

```
# Directive: enter text section
#
#
#
# allocate stack frame
# save return address
# compute address of string1
#
# compute address of string2
#
# call function printf
# restore return address
# deallocate stack frame
# load return value 0
# return
#
#
# label for first string
#
# label for second string
```

Compiled Hello.c: Hello.s

```
     .text
     main:
0x00   addi sp,sp,-16
0x04   sw   ra,12(sp)
0x08   lui  a0,???
0x0c   addi a0,a0,???
0x10   lui  a1,???
0x14   addi a1,a1,???
0x18   jal, ra, ???
0x1c   lw   ra,12(sp)
0x20   addi sp,sp,16
0x24   addi a0, a0, 0
0x28   jalr, x0, ra, 0

     .data
0x00 str1:
       .string "Hello, %s!\n"
0x0c str2:
       .string "world"
```

Symbol Table:

Label:	Offset:	Type:
main	0x00000000	global text
str1	0x00000000	local data
str2	0x0000000c	local data

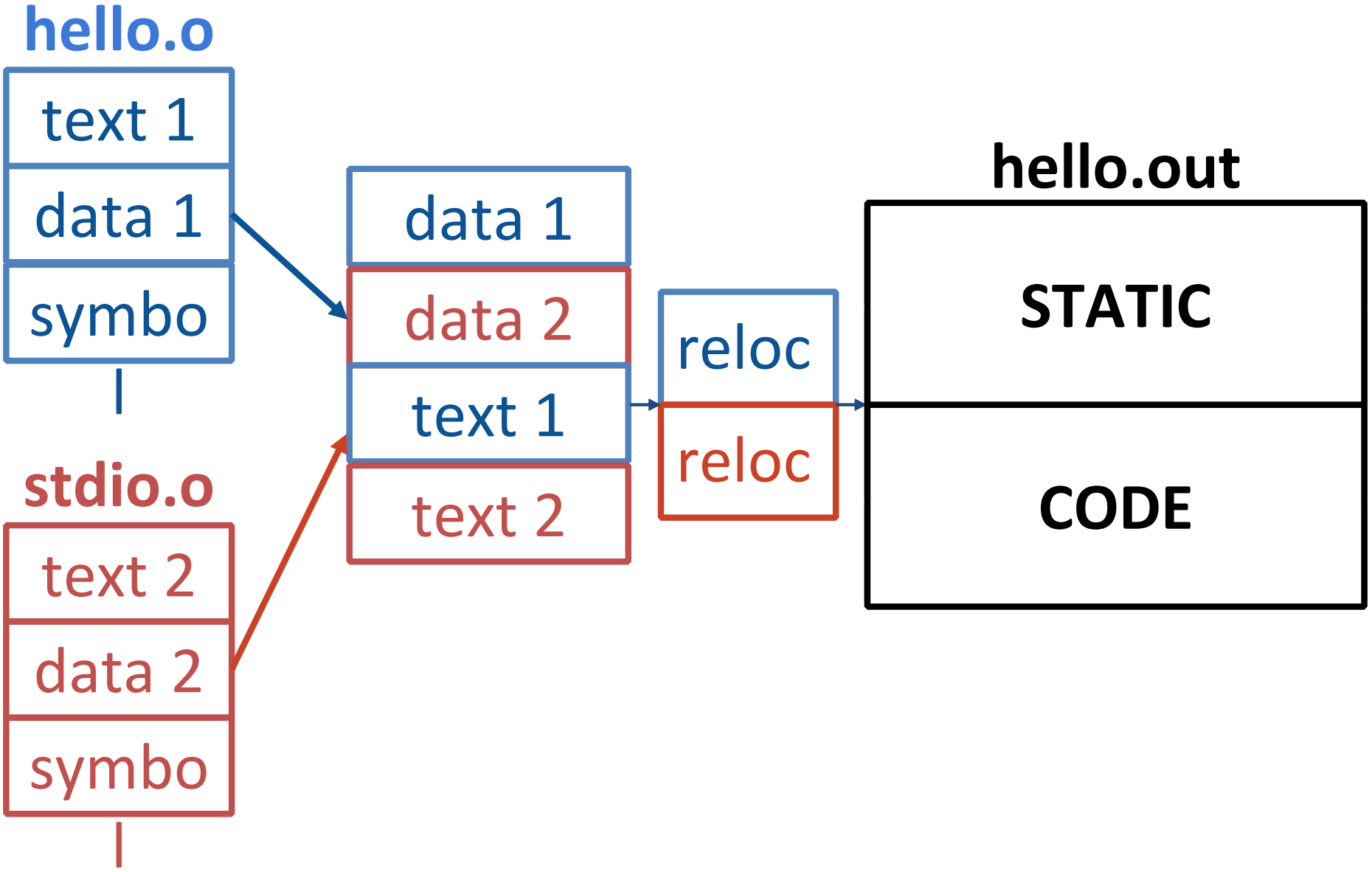
Relocation Table:

Offset:	Inst:	Dependency:
0x00000008	lui	str1
0x0000000c	addi	str1
0x00000010	lui	str2
0x00000014	addi	str2
0x00000018	jal	printf

Assembled Hello.s → Hello.o

```
00000000 <main>:
00: ff010113 addi sp,sp,-16
04: 00112623 sw ra,12(sp)
08: 00000537 lui a0,0x0     # addr placeholder
0c: 00050513 addi a0,a0,0   # addr placeholder
10: 000005b7 lui a1,0x0     # addr placeholder
14: 00058593 addi a1,a1,0   # addr placeholder
18: 000000ef jal ra,0x0     # addr placeholder
1c: 00c12083 lw ra,12(sp)
20: 01010113 addi sp,sp,16
24: 00000513 addi a0,a0,0
28: 00008067 jalr x0, ra, 0
```

Linker


hello.o
text 1
data 1
symbo
stdio.o
text 2
data 2
symbo
data 1
data 2
text 1
text 2
reloc
reloc
hello.out
STATIC
CODE

Linked Text

```
000101b0 <main>:
   101b0: ff010113 addi sp,sp,-16
   101b4: 00112623 sw   ra,12(sp)
   101b8: 00021537 lui  a0,0x21
   101bc: a1050513 addi a0,a0,-1520 #20a10<str1>
   101c0: 000215b7 lui  a1,0x21
   101c4: a1c58593 addi a1,a1,-1508 #20a1c<str2>
   101c8: 288000ef jal  ra,10450    #<printf>
   101cc: 00c12083 lw   ra,12(sp)
   101d0: 01010113 addi sp,sp,16
   101d4: 00000513 addi a0,0,0
   101d8: 00008067 jalr  ra
```

Loader