System-Level Programming

34 Organisation of Memory

J. Kleinöder, D. Lohmann, V. Sieh, P. Wägemann

Lehrstuhl für Informatik 4
Systemsoftware

Friedrich-Alexander-Universität
Erlangen-Nürnberg

Summer Term 2024

http://sys.cs.fau.de/lehre/ss24

Organisation of Memory

```
int a;                          // a: global, uninitialized
int b = 1;                      // b: global, initialized
const int c = 2;                // c: global, const

void main(void) {
    static int s = 3;           // s: local, static, initialized
    int x, y;                   // x: local, auto; y: local, auto
    char *p = malloc(100);      // p: local, auto; *p: heap (100 byte)
}
```

Where does the memory for these variables come from?

Organisation of Memory

```
int a;                          // a: global, uninitialized
int b = 1;                      // b: global, initialized
const int c = 2;                // c: global, const

void main(void) {
    static int s = 3;           // s: local, static, initialized
    int x, y;                   // x: local, auto; y: local, auto
    char *p = malloc(100);      // p: local, auto; *p: heap (100 byte)
}
```

Where does the memory for these variables come from?

- **Static allocation** – allocation during compilation / linking
  - Concerns all global/static variables and the code itself ↪ 12–5
  - Allocation by getting placed into a section

.text	– contains program code	`main()`
.bss	– contains all variables initialized with 0	`a`
.data	– contains all variables initialized with other values	`b,s`
.rodata	– contains all constant variables	`c`

Organisation of Memory

```c
int a;                          // a: global, uninitialized
int b = 1;                      // b: global, initialized
const int c = 2;                // c: global, const

void main(void) {
    static int s = 3;           // s: local, static, initialized
    int x, y;                   // x: local, auto; y: local, auto
    char *p = malloc(100);      // p: local, auto; *p: heap (100 byte)
}
```

Where does the memory for these variables come from?

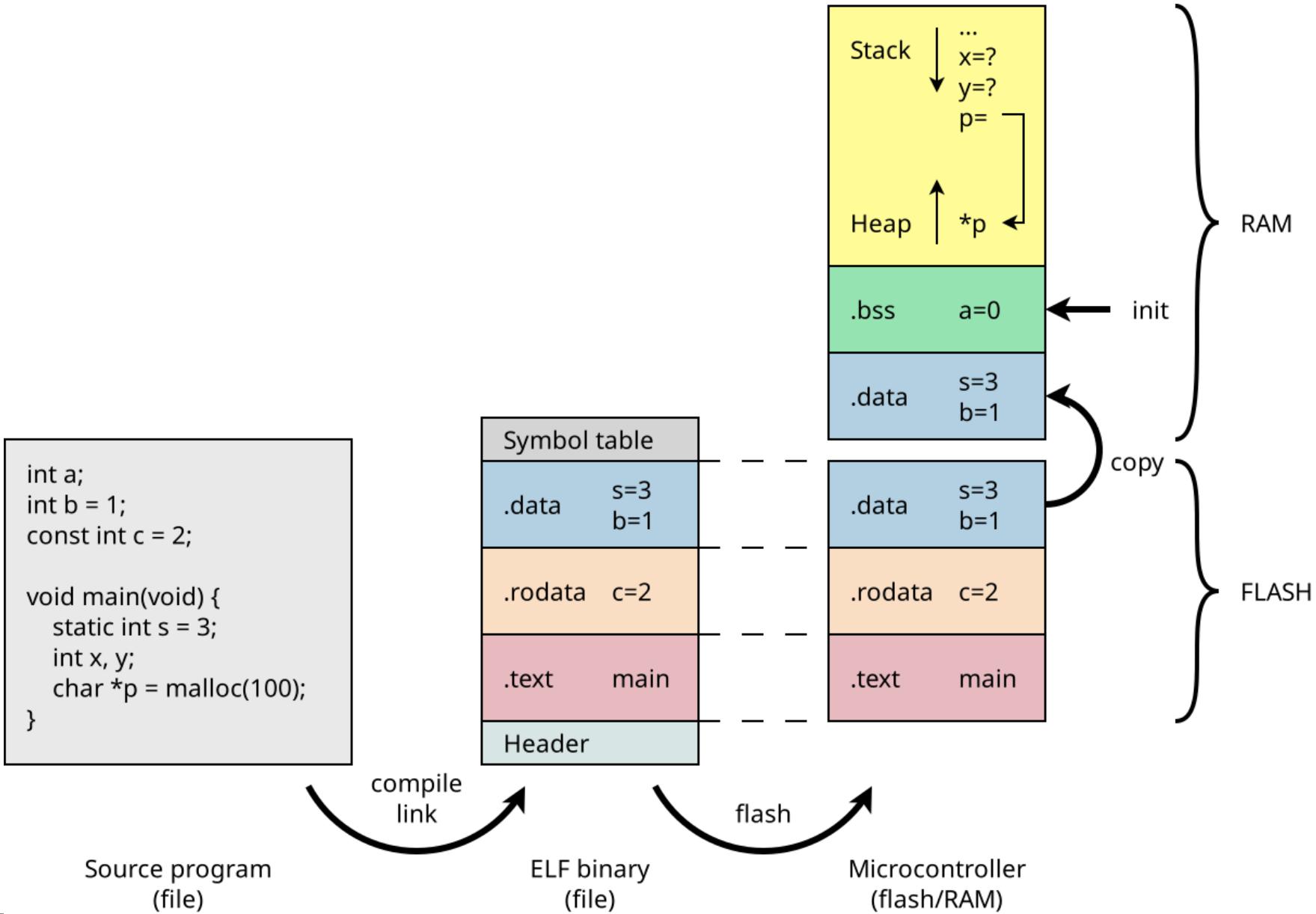
- **Static allocation** – allocation during compilation / linking
  - Concerns all global/static variables and the code itself ↪ 12–5
  - Allocation by getting placed into a section

.text	– contains program code	`main()`
.bss	– contains all variables initialized with 0	`a`
.data	– contains all variables initialized with other values	`b,s`
.rodata	– contains all constant variables	`c`

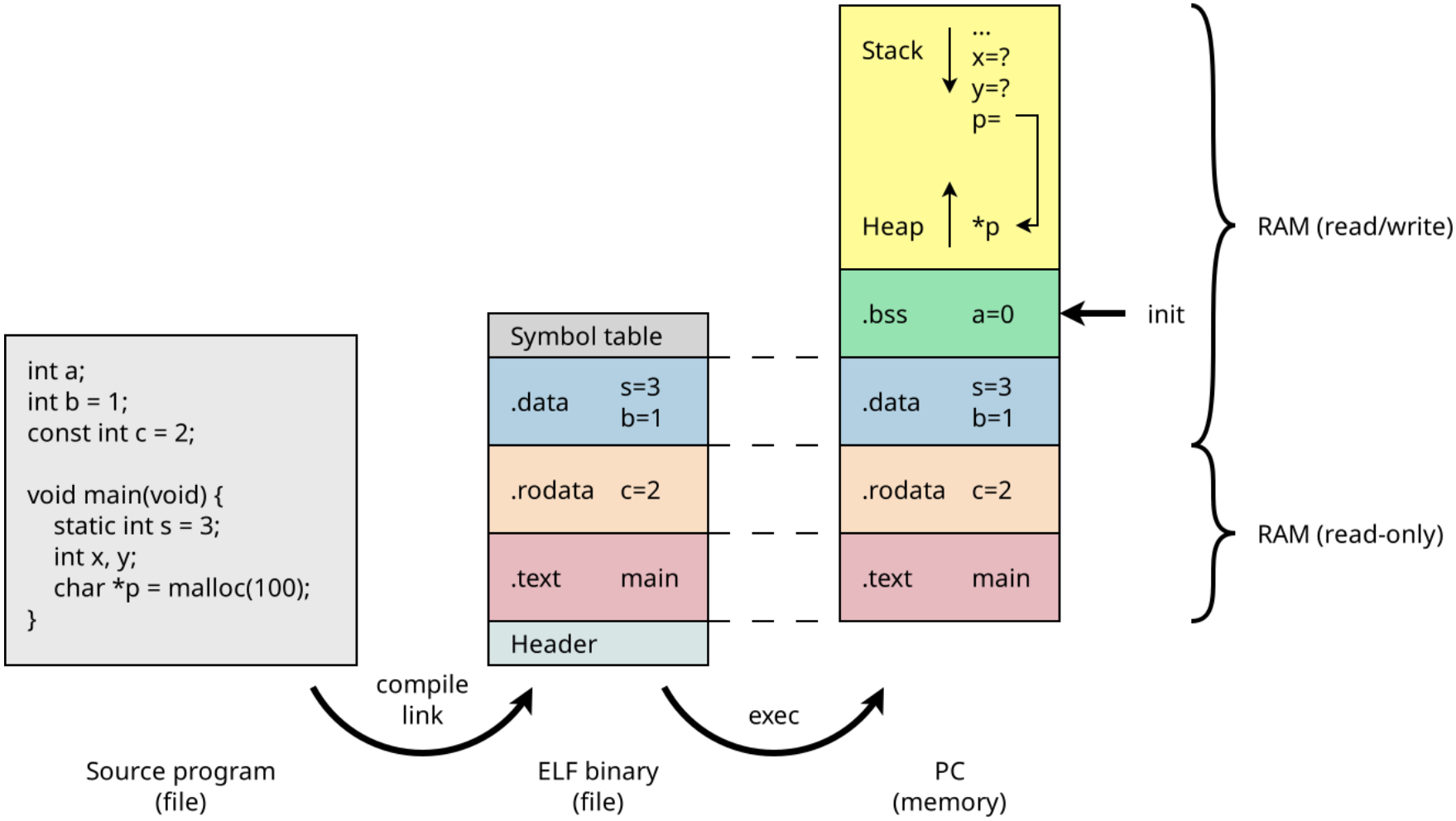
- **Dynamic allocation** – reserved during runtime
  - Concerns all local automatic variables and explicitly allocated memory

Stack	– contains all auto variables that are currently alive	`x,y,p`
Heap	– contains with `malloc()` explictly allocated memory areas	`*p`

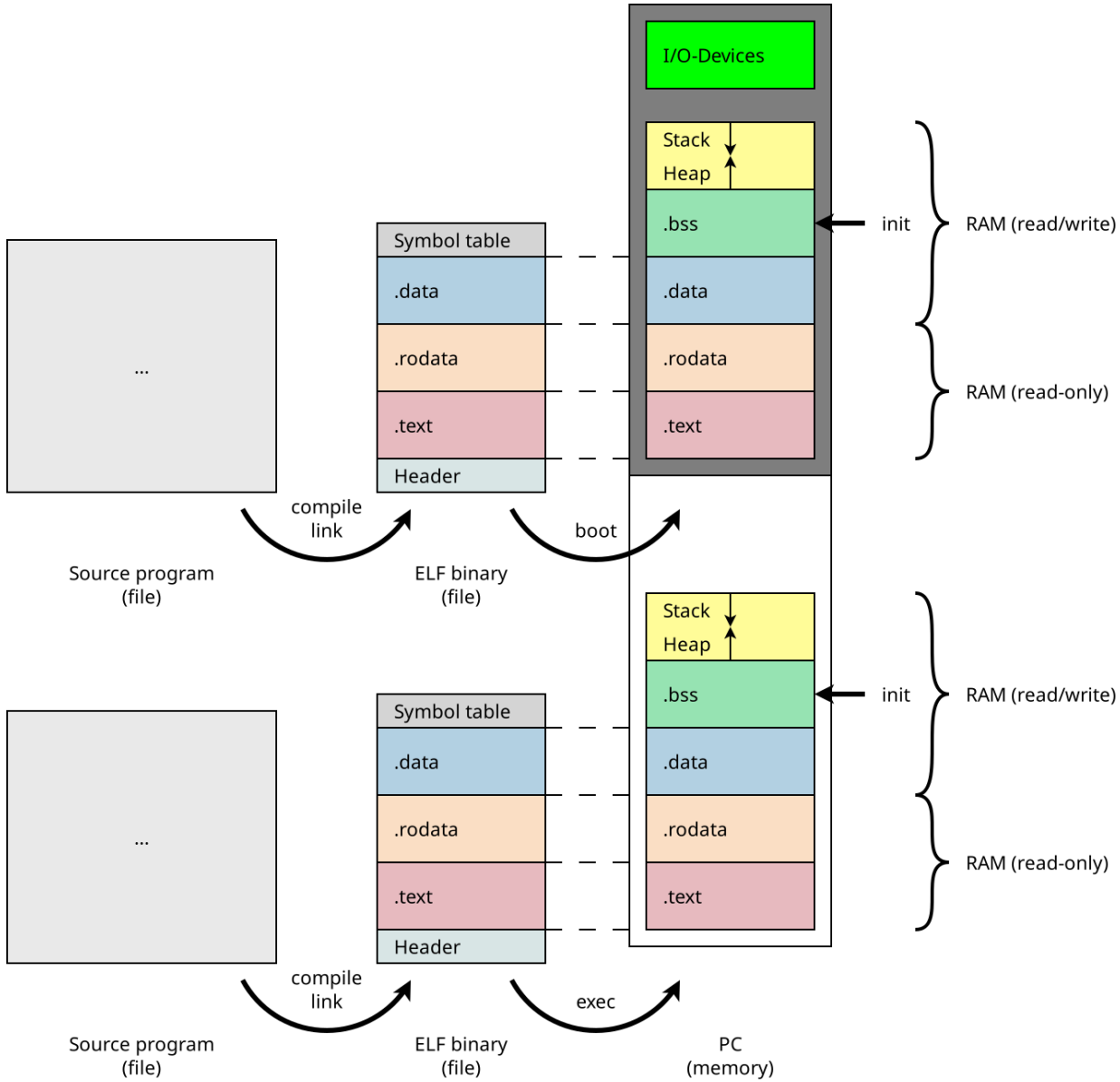
Organisation of Memory on a μC



Organisation of Memory with an OS



Organisation of Memory with an OS (continued)



Organisation of Memory with an OS (continued)