

A Tour of Computer Systems

Computer system

- Consists of hardware and systems software that work together to run application programs

```
1  #include <stdio.h>
2
3  int main()
4  {
5      printf("hello, world\n");
6      return 0;
7  }
```

The ASCII text representation of hello.c

#	i	n	c	l	u	d	e	SP	<	s	t	d	i	o	.
35	105	110	99	108	117	100	101	32	60	115	116	100	105	111	46
h	>	\n	\n	i	n	t	SP	m	a	i	n	()	\n	{
104	62	10	10	105	110	116	32	109	97	105	110	40	41	10	123
\n	SP	SP	SP	SP	p	r	i	n	t	f	("	h	e	l
10	32	32	32	32	112	114	105	110	116	102	40	34	104	101	108
l	o	,	SP	w	o	r	l	d	\	n	")	;	\n	SP
108	111	44	32	119	111	114	108	100	92	110	34	41	59	10	32
SP	SP	SP	r	e	t	u	r	n	SP	0	;	\n	}	\n	
32	32	32	114	101	116	117	114	110	32	48	59	10	125	10	

Programs Are Translated by Other Programs into Different Forms

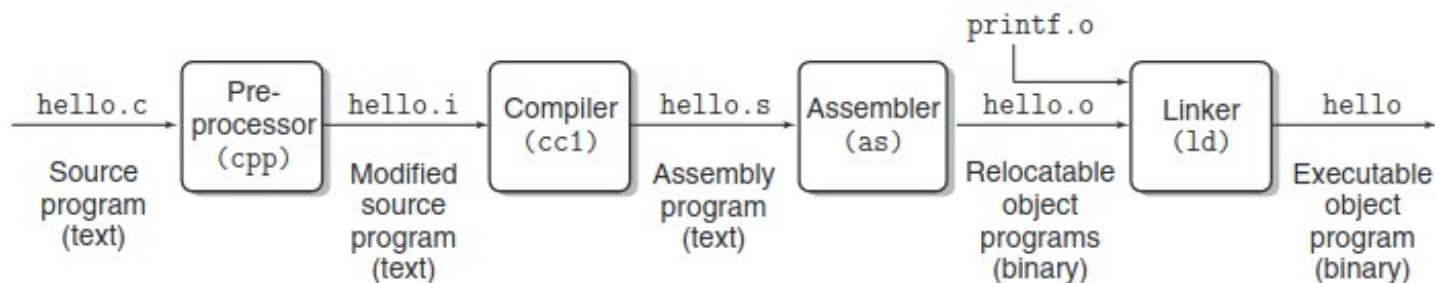


Figure 1.3 The compilation system.

translation from source file to object file is performed

```
linux> gcc -o hello hello.c
```

by a compiler driver:

- **Translation from source file to object file is performed by a compiler**
 - gcc compiler driver reads the source file `hello.c` and translates it into an executable object file `hello`.
- **The translation is performed in the sequence of four phases**
- **The programs that perform the four phases (preprocessor, compiler, assembler, and linker)**

- **Preprocessing Phase:**

- The preprocessor (cpp) modifies the original C program according to directives that begin with the '#' character
- The result is another C program, typically with the .i suffix.(hello.i)

- **Compilation phase.**

- The compiler (cc1) translates the text file hello.i into the text file hello.s
- I.e it contains an assembly-language program

```
1  main:
2      subq    $8, %rsp
3      movl    $.LC0, %edi
4      call    puts
5      movl    $0, %eax
6      addq    $8, %rsp
7      ret
```

- **Assembly phase.**

- The assembler (as) translates hello.s into machine-language instructions, packages them in a form known as object program, and stores the result in the object file hello.o.

- **Linking phase.:**

- The process of rearranging the existing code and filling missed code.
- Example : The printf function(which is part of the standard C library) resides in a separate precompiled object file called printf.o, which must somehow be merged with our hello.o program.

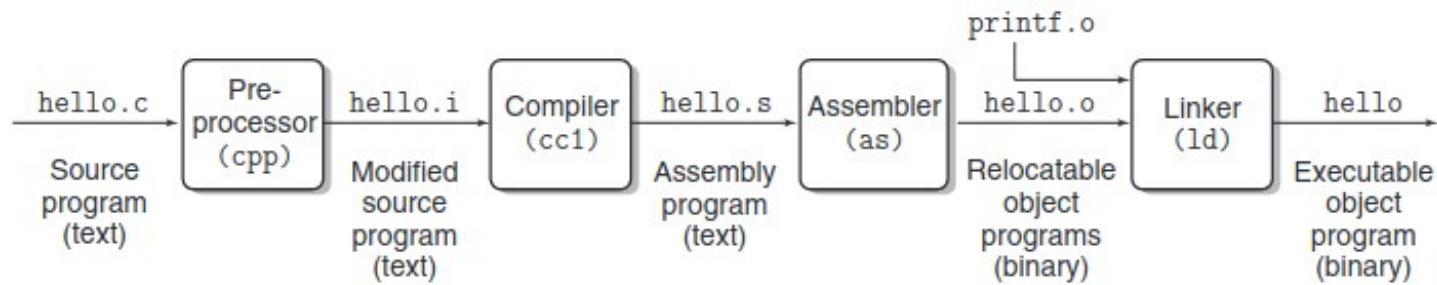


Figure 1.3 The compilation system.

```
linux> gcc -o hello hello.c
```

- **Preprocessing**

- `cc -E abc.c`

- **Compilation**

- `cc -S abc.c`
 - Output : `.s` file

-

- **Assembly**

- `cc -c abc.c`
 - Output : `.o` file

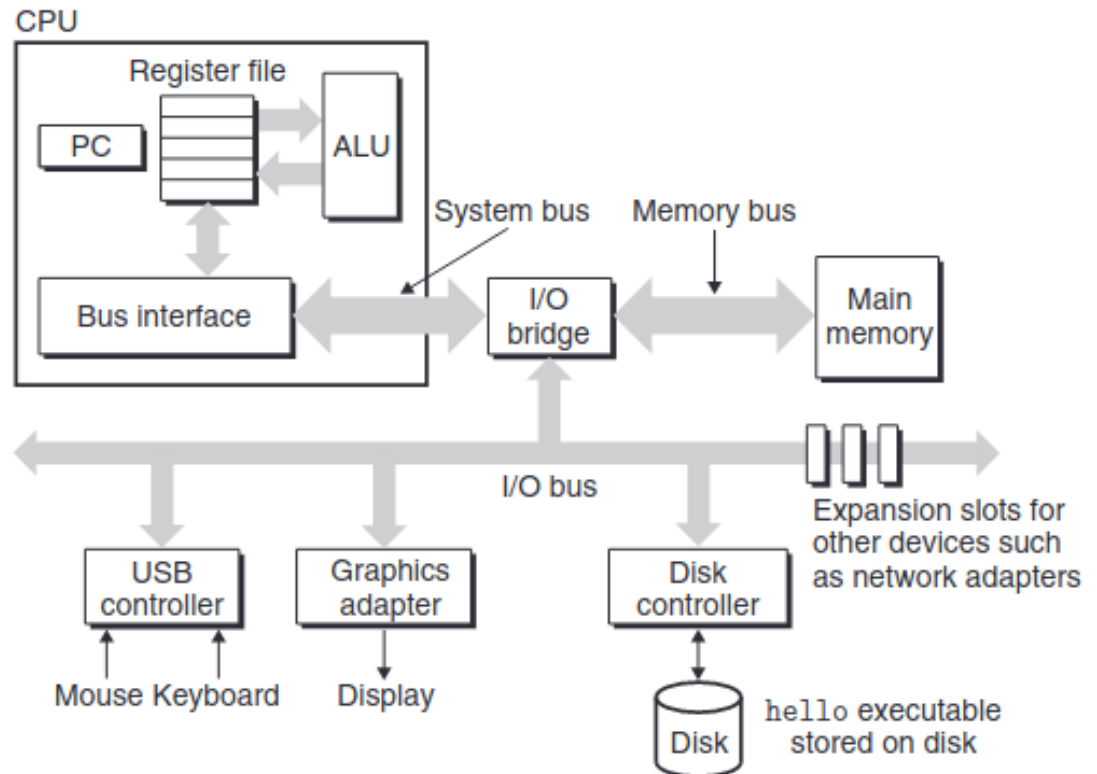
- **Linking**

- `cc -o abc abc.c`

Processors Read and Interpret Instructions Stored in Memory

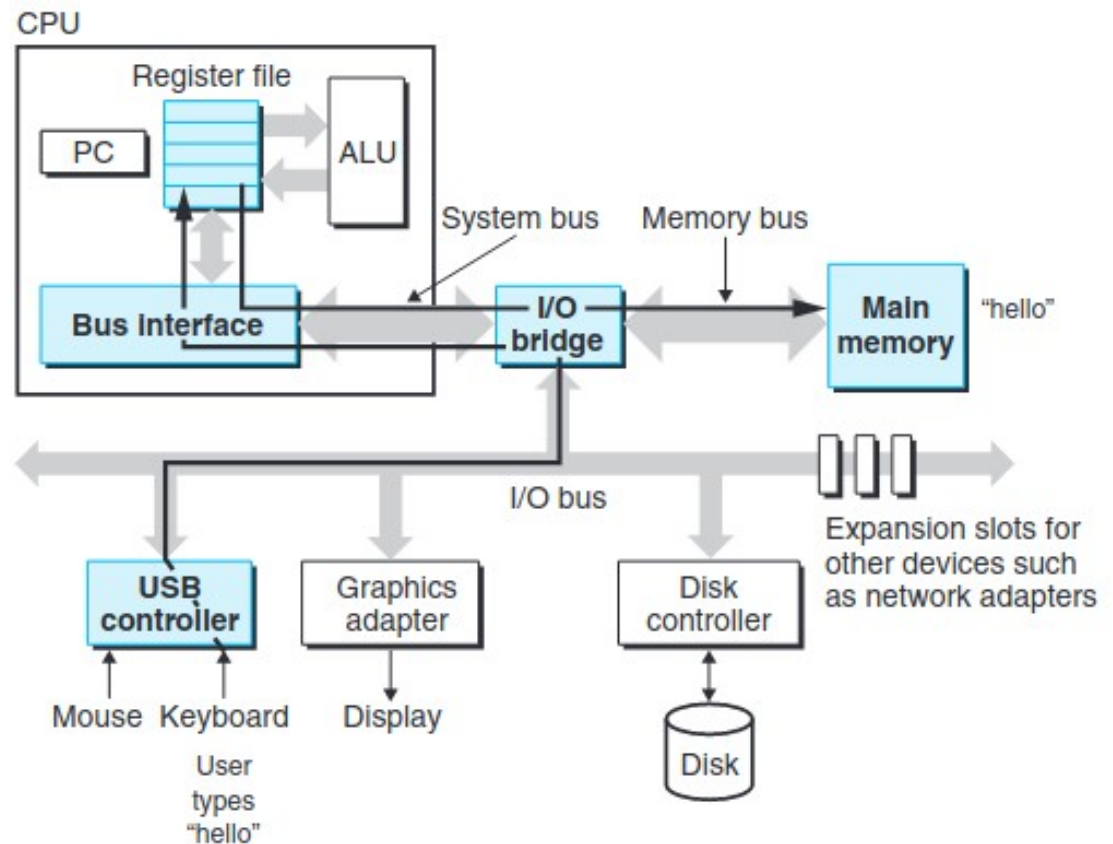
- **Hardware Organization of a System**

- Buses
- I/O Devices
- Main Memory
- Processor



Running the hello Program

```
:~$ cc hello.c  
:~$ ./a.out  
Hello, World!
```



Memory hierarchy.

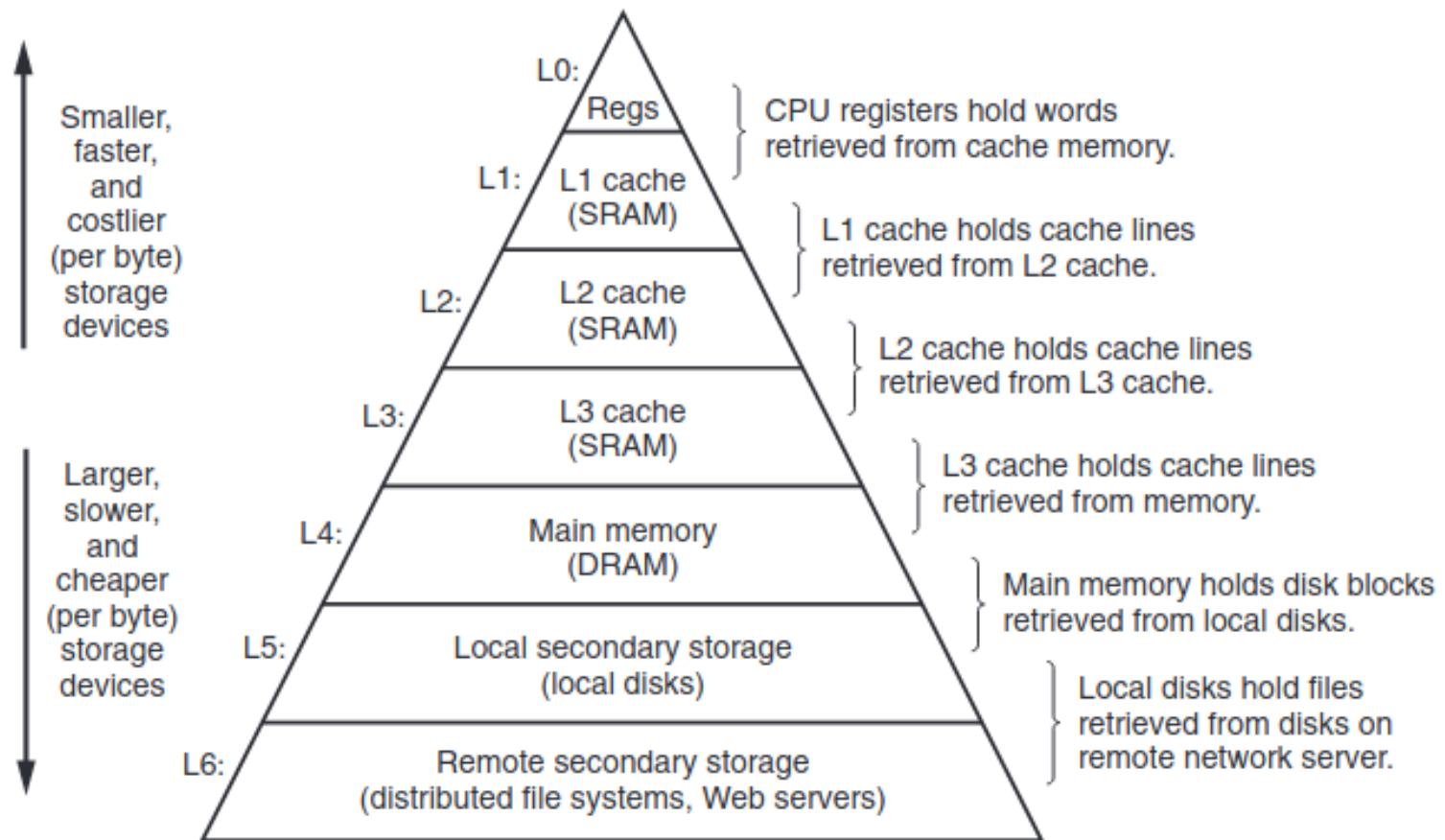


Figure 1.9 An example of a memory hierarchy.