Where are We Going?

Processor and memory architecture

Peripherals: GPIO, timers, UART

Assembly language and machine code

From C to assembly language

Functions and stack frames

Serial communication and strings

Modules and libraries: Linking

Memory Map: Loading, starting, and the heap

gpio
timer
uart
printf
malloc
keyboard
mailbox
fb
gl
console
monitor

blink/

Why Modules?

Logical decomposition of the system into parts

Interfaces and implementations

- **Clean and easy-to-use abstractions**
- **Hide obscure details**
- **Isolate consumers from providers**

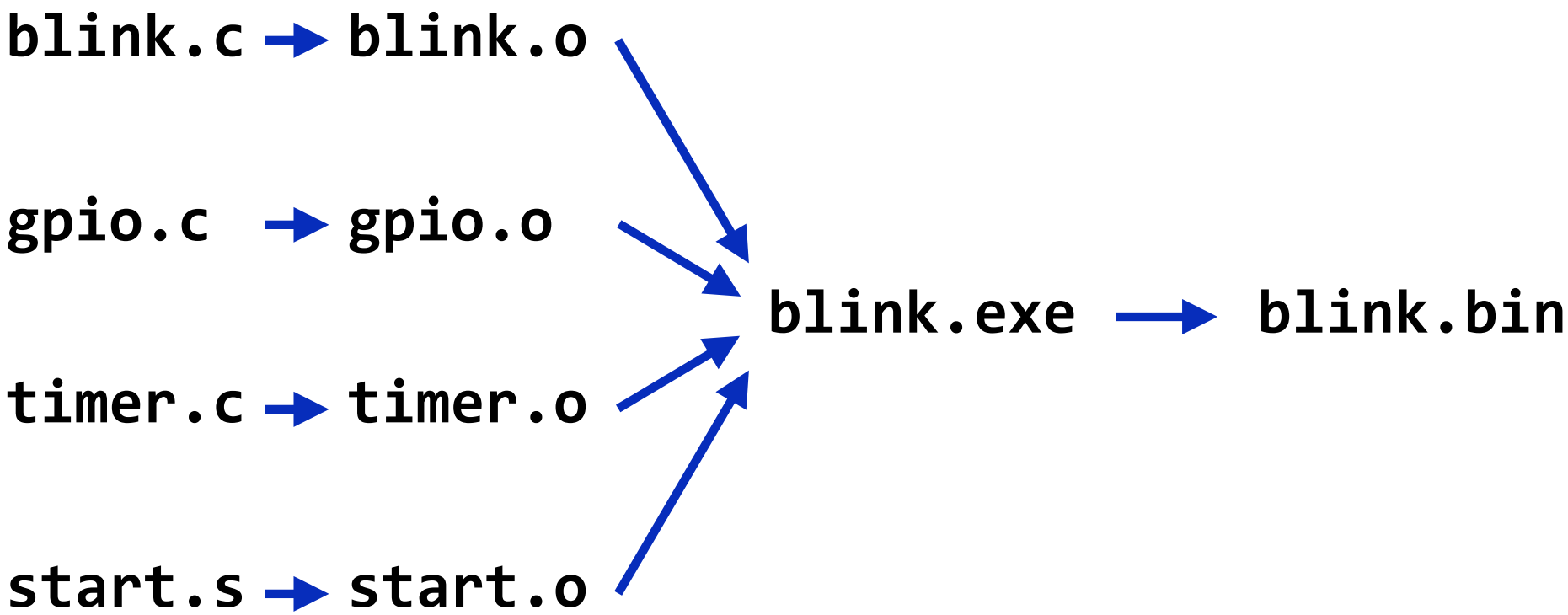
Tested independently with unit tests

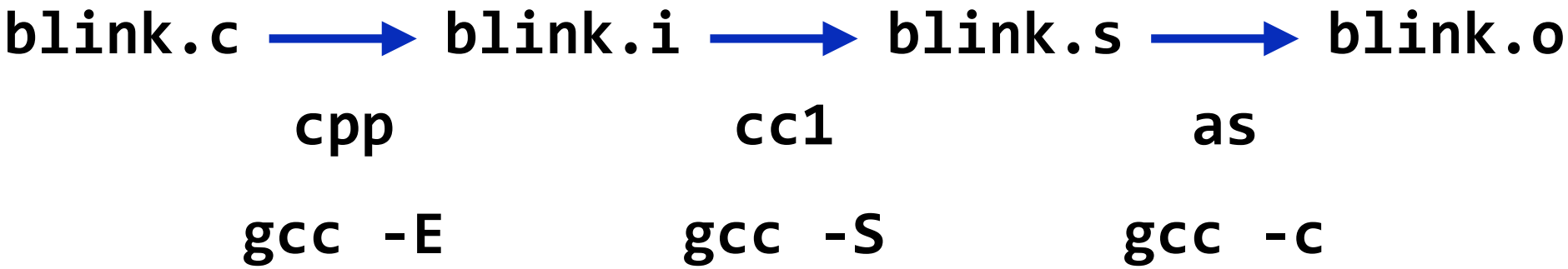
Obi-wan, how can I design powerful modules?

blink/Makefile

How to Build Parts?

```
NAME = blink
OBJECTS = gpio.o timer.o start.o

$(NAME).exe: $(NAME).o $(OBJECTS)
    arm-none-eabi-ld $(LDFLAGS)  $^ -o $@

%.o: %.c
    arm-none-eabi-gcc $(CFLAGS) -c $< -o $@

%.o: %.s
    cpp -P $< | arm-none-eabi-as $(ASFLAGS) -o $@

%.list: %.o
    arm-none-eabi-objdump -d $< > $@

%.bin: %.exe
    arm-none-eabi-objcopy $< -O binary $@
```


blink.c
blink.o
gpio.c
gpio.o
timer.c
timer.o
start.s
start.o
blink.exe
blink.bin



gcc --savetemps

When to Build Parts?

Build target if it does not exist

Rebuild target if any of the dependencies have changed (are newer than the target)

WARNING: only considers explicit dependencies; there may be other dependencies

Linking

`blink/`

Symbols

Definitions: global vs local (`static`)

- **by default symbols are local in `.s` files**
- **by default symbols are global in `.c` files**

Defined vs. undefined (`extern`)

Single global name space

- **need `gpio_` prefix to prevent collisions**

Local variables in functions are not symbols

Symbol Resolution

Set of defined symbols D

Set of undefined symbols U

Moving left to right, for each .o file, the linker updates U and D and proceeds to next input file.

Problems

- **If two files try to define the same symbol, an error is reported (unless it is a COMMON).**
- **After all the input arguments are processed, if U is found to be not empty, the linker prints an error report and terminates.**


Intermission

Questions?

Suppose you edit your Makefile, what kinds of changes would require rebuilding the `.o`'s

Suppose you change the interface in a header file, what do you need to rebuild?

When to Build?

change implementation (`gpio.c`)?

- **recompile implementation (`gpio.c`)**

change interface (`gpio.h/gpio.c`)?

- **must recompile clients of the interface (`blink.c`)**
- **`blink.c` depends on `gpio.h`**

linking

```
// uninitialized global and static
int i;
static int j;

// initialized global and static
int k = 1;
static int l = 2;

// initialized global and static const
const int m = 3;
static const int n = 4;
```

```
% arm-none-eabi-nm -S tricky.o
00000004 00000004 C i
00000000 00000004 b j
00000000 00000004 D k
00000004 00000004 d l
00000000 00000004 R m
                  U p
00000000 00000048 T tricky

# The global uninitialized variable i
# is in common (C).

# The static const variable n
# has been optimized out.
```

Symbols

Types

- extern (undefined)
- global vs static
- initialized vs uninitialized
- const vs non-const

Guide to Symbols

T/t - text

D/d - read-write data

R/r - read-only data

B/b - bss (*Block Started by Symbol*)

C - common (instead of B)

lower-case letter means `static`

Sections

Instructions go in `.text`

Data goes in `.data`

`const` data (read-only) goes in `.rodata`

Uninitialized data goes in `.bss`

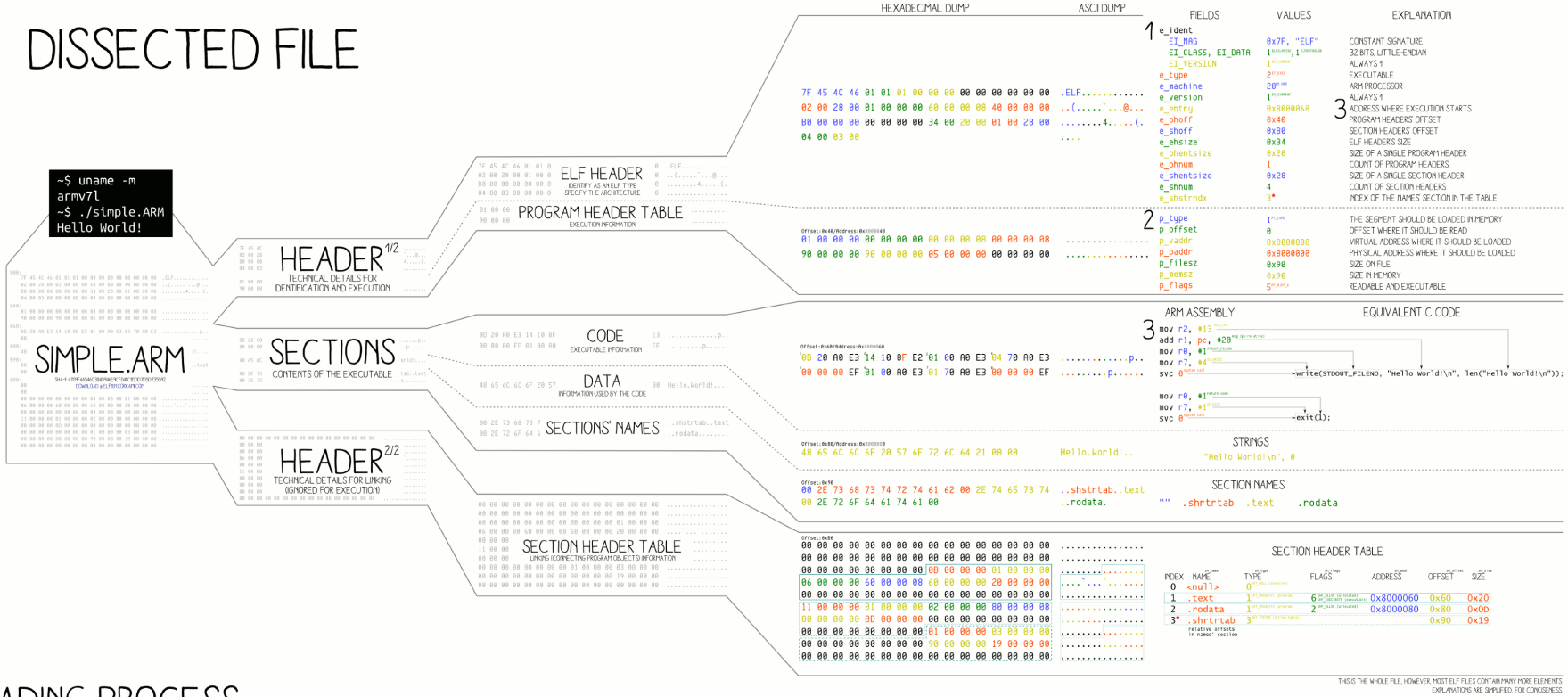
+ other information about the program

- **symbols, relocation, debugging, ...**

ELF[101] a Linux executable walk-through

ANGE ALBERTINI
CORKAMI.COM

DISSECTED FILE



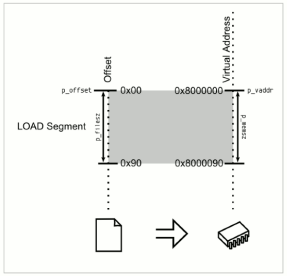
LOADING PROCESS

1 HEADER

THE ELF HEADER IS PARSED
THE PROGRAM HEADER IS PARSED
(SECTIONS ARE NOT USED)

2 MAPPING

THE FILE IS MAPPED IN MEMORY
ACCORDING TO ITS SEGMENT(S)



3 EXECUTION

ENTRY IS CALLED
SYSCALLS[GLOSS] ARE ACCESSED VIA:
- SYSCALL NUMBER IN THE R7 REGISTER[ARM]
- CALLING INSTRUCTION SVC

TRIVIA

THE ELF WAS FIRST SPECIFIED BY U.S.L. AND U.I.[SYSTEM INTERNATIONAL]
FOR UNIX SYSTEM V, IN 1989

THE ELF IS USED, AMONG OTHERS, IN:
- LINUX, ANDROID, *BSD, SOLARIS, BEOS
- PSP, PLAYSTATION 2-4, DREAMCAST, GAMECUBE, WII
- VARIOUS OSES MADE BY SAMSUNG, ERICSSON, NOKIA,
- MICROCONTROLLERS FROM ATMEL, TEXAS INSTRUMENTS

VERSION 1.0A
2013/12/06

Libraries

`libpi/`

Symbols from in Libraries

An archive `.a` is just a collection of `.o` files

The linker scans the library for any `.o` files that contain definitions of undefined symbols. If a file in the library contains an undefined symbol, it is included in the linker file list

Adding the `.o` file from the library may result in more undefined symbols; the linker searches for the definition of these symbols in the library and includes the relevant files; this search is repeated until no more definitions of undefined symbols are found

Development Tasks

Managing changes to the source

Compiling pieces individually

Combining ("linking") the pieces together into a single executable program

Building the systems quickly and reliably

- **Quickly means incrementally**
- **Reliably means rebuilding modules when necessary**

Your Tools are "*Special*"

Programs
That Manipulate
Programs