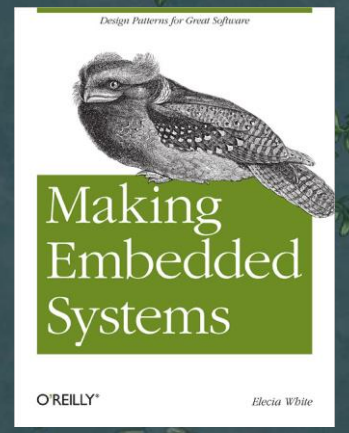
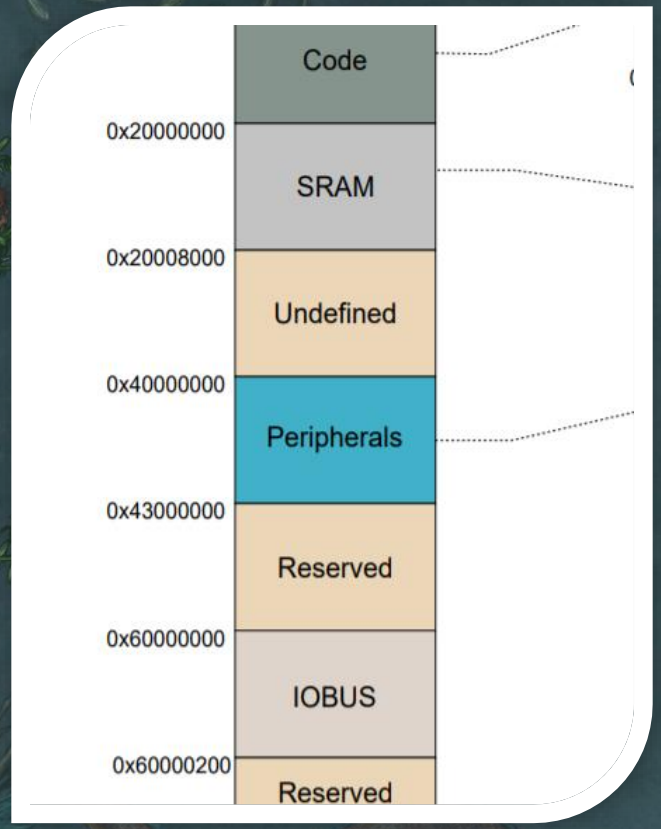
Map Files and Other Buried Treasures

Heaps and Stacks and Memory Maps
But Definitely *Not* Linker Files



Memory Maps

There are many ways of looking at memory

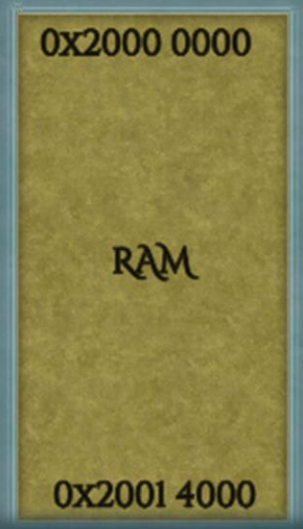


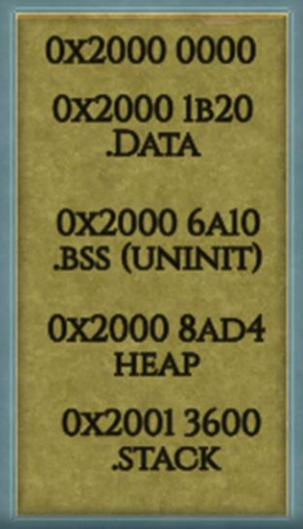
1		Start	Size
2	Flash Memory	0x0000 0000	0x58000
3	RAM	0x2000 0000	0x14000
4			
5	Flash	Start	Size
6	Image Header	0x0000 0000	0xCF
7	Code	0x0000 00D0	0x474bc
8	NV Storage	0x0005 2000	0x3FFF
9	Bootloader	0x0005 6000	0x1FFF
10	Flash End	0x0005 8000	

Elecia White, Logical Elegance, Inc. https://embedded.fm/blog/MapFiles

Memory Layout

Planning where things go





Use the Map File

Problem	Map Tool
Not enough RAM	Look at summary
Not enough code space	Diff with good map file
Hard fault errors	Find/write viewer
Weird memory errors	Search for address nearby
Planning FW update	Search for variable name
Running too slow	Statistical sampling (hard)
	Read each and every line

Foreshadowing…

Look at Hello.map

TI CCS, CC26XR1

Example hello: prints out "Hello World" to UART

Uses TI's RTOS

Your .map is probably located where your .hex file is

A More Complicated Map File

Hello was 2162 lines long

This one is 14034 lines long

Both TI CCS

A Real Memory Map

Ooooh…. I love this part

Use the Map File

Problem	Map Tool
Not enough RAM	Look at summary
Not enough code space	Diff with good map file
Hard fault errors	Find/write viewer
Weird memory errors	Search for address nearby
Planning FW update	Search for variable name
Running too slow	Statistical sampling (hard)
	Read each and every line

Not every tool works for every problem.

Use the Map File

Problem	Map Tool
Not enough RAM	Look at summary
Not enough code space	Diff with good map file
	Find/write viewer
	Search for address nearby
	Search for variable name
	Statistical sampling (hard)
	Read each and every line

If the map is a wall of impenetrable text, choose a (non-static) global variable or function, one you know is large, and search for it in the map file.

Use the Map File

Problem

Not enough RAM

Not enough code space

Map Tool

Look at summary

Diff with good map file

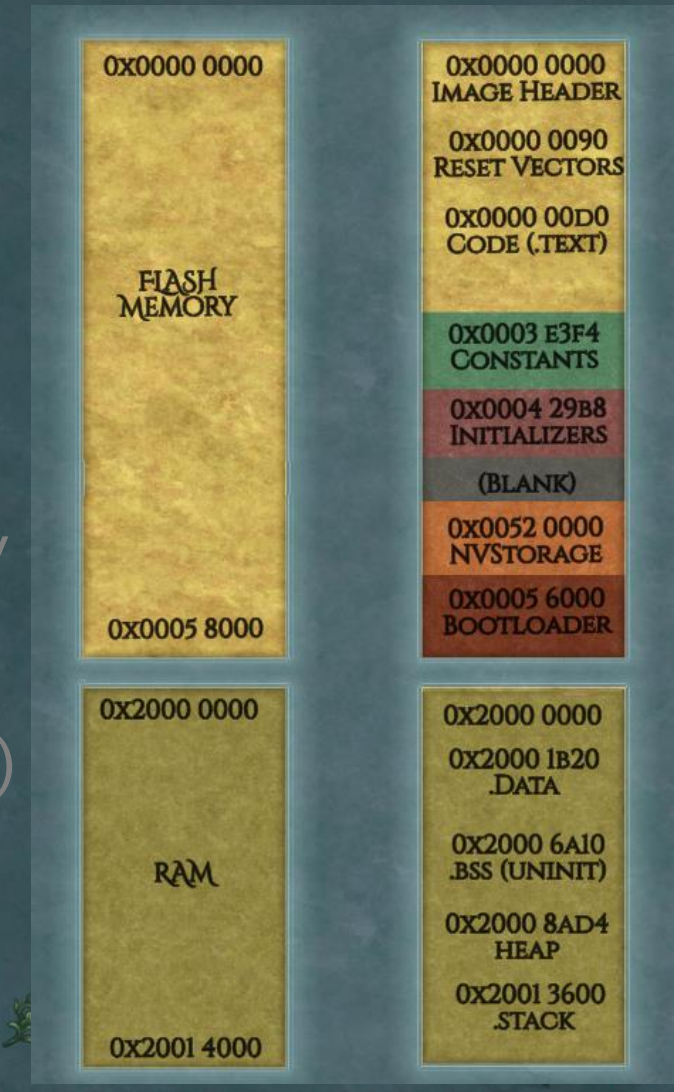
Find/write viewer

Search for address nearby

Search for variable name

Statistical sampling (hard)

Read each and every line



Space Optimization Scorecard

Action	Text (code)	Data	Total	Total (hex)	Freed	Total freed
Baseline	31949	324	32273	7E11		
Commented-out test code	26629	324	26953	6949	5320	(Reverted change)
Reimplemented abs()	29845	324	30169	75D9	2104	2104
Calculated const table at init time	29885	244	30129	75B1	40	2144
= comment from you	= size of .text section	= size of .data section	= total image size	= hex of total image size	= bytes freed with this change	= total bytes freed since start

Use the Map File

Problem	Map Tool
Not enough RAM	Look at summary
Not enough code space	Diff with good map file
	Find/write viewer
	Search for address nearby
	Search for variable name
	Statistical sampling (hard)
	Read each and every line

If the map is a wall of impenetrable text, choose a (non-static) global variable or function, one you know is large, and search for it in the map file.

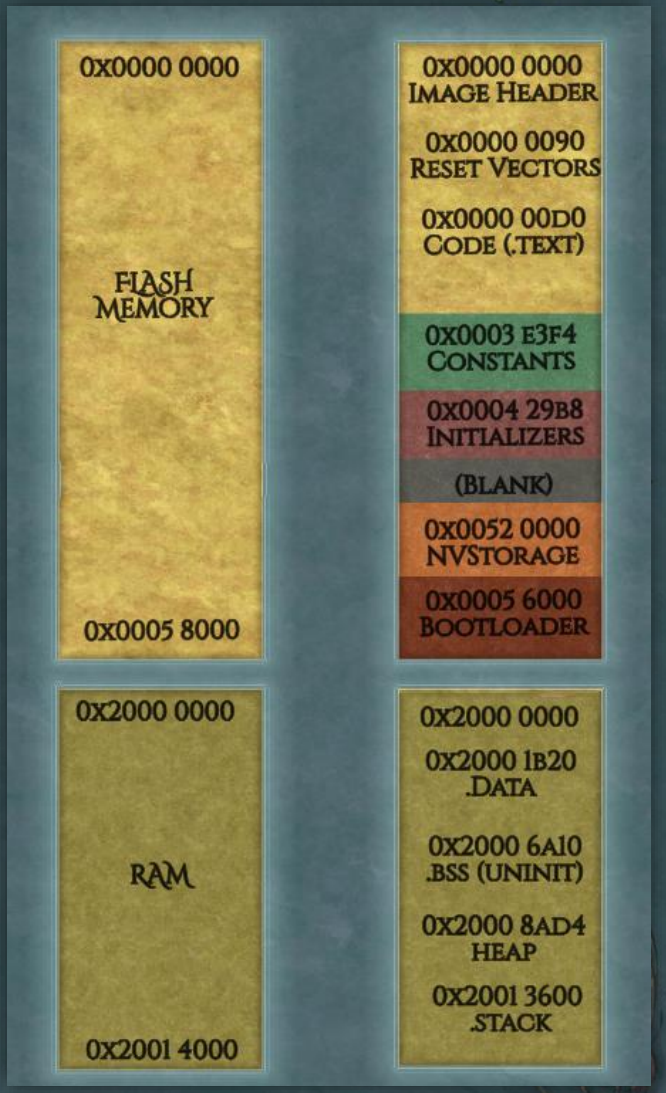
But how much RAM do I have left?

Just look in the memory configuration mountains! The summary section lists how much memory you and you can see how much you have left!

Easy peasy!

```
10 MEMORY CONFIGURATION
11
12          name             origin    length      used     unused   attr    fill
13 ----------------------   --------  ---------  --------  --------  ----  --------
14   FLASH                  00000000   00058000  000030cb  00054f35  R  X
15   GPRAM                  11000000   00002000  00000000  00002000  RW X
16   SRAM                   20000000   00014000  000020af  00011f51  RW X
17
```

Asking more will lead to trouble.



Zero bytes left!

RAM ALLOCATION

CREATED BY ELECIA WHITE , LOGICAL ELEGANCE, INC.

7 + 5 + 3 = 15
15 corals available
or maybe zero due to
memory fragmentation

RAM ALLOCATION

CREATED BY ELECIA WHITE , LOGICAL ELEGANCE, INC.

“

The answer ‘I don’t know’ is very unsatisfying.

”

-Me

Don’t use malloc. Don’t use recursion. Don’t do anything fun.

Use the Map File

Problem	Map Tool
	Look at summary
	Diff with good map file
Hard fault errors	Find/write viewer
Weird memory errors	Search for address nearby
	Search for variable name
	Statistical sampling (hard)
	Read each and every line

Let's talk about debugging the impossible bugs.

You know, those icky, crawly ones that you worry about but can't reliably reproduce.

Walk me through it from power on to where the error occurred…

- Is it powered? Are you sure?
- Are you sure that is it running the code you think it is?
- Can you test only that part of the system?
- Did you check the errata for the part?
- If it is intermittent, can you make it happen more often?
- Is it a timing error? Stack overflow? Uninitialized variable?
- Can you turn optimizations off and see if it still happens?
- A variable that isn't a volatile but should be?
- Have you looked at the map file?

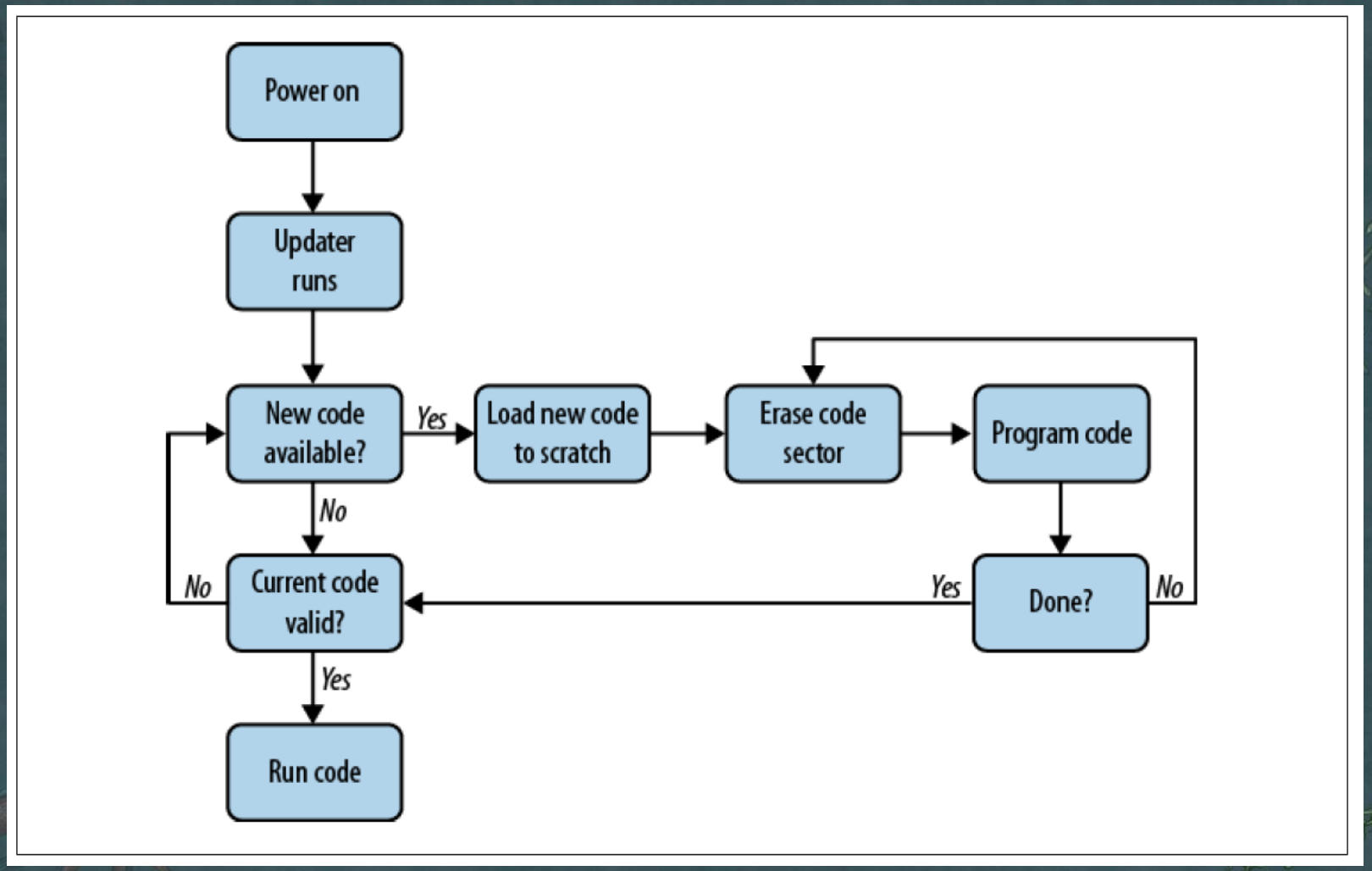
- **In case of emergency (and random nonsensical error): could it be a ground loop problem?**

Use the Map File

Problem	Map Tool
	Look at summary
	Diff with good map file
	Find/write viewer
	Search for address nearby
Planning FW update	Search for variable name
	Statistical sampling (hard)
	Read each and every line

Where, exactly, did I leave the bootloader?

Firmware Update



Elecia White, Logical Elegance, Inc. https://embedded.fm/blog/MapFiles

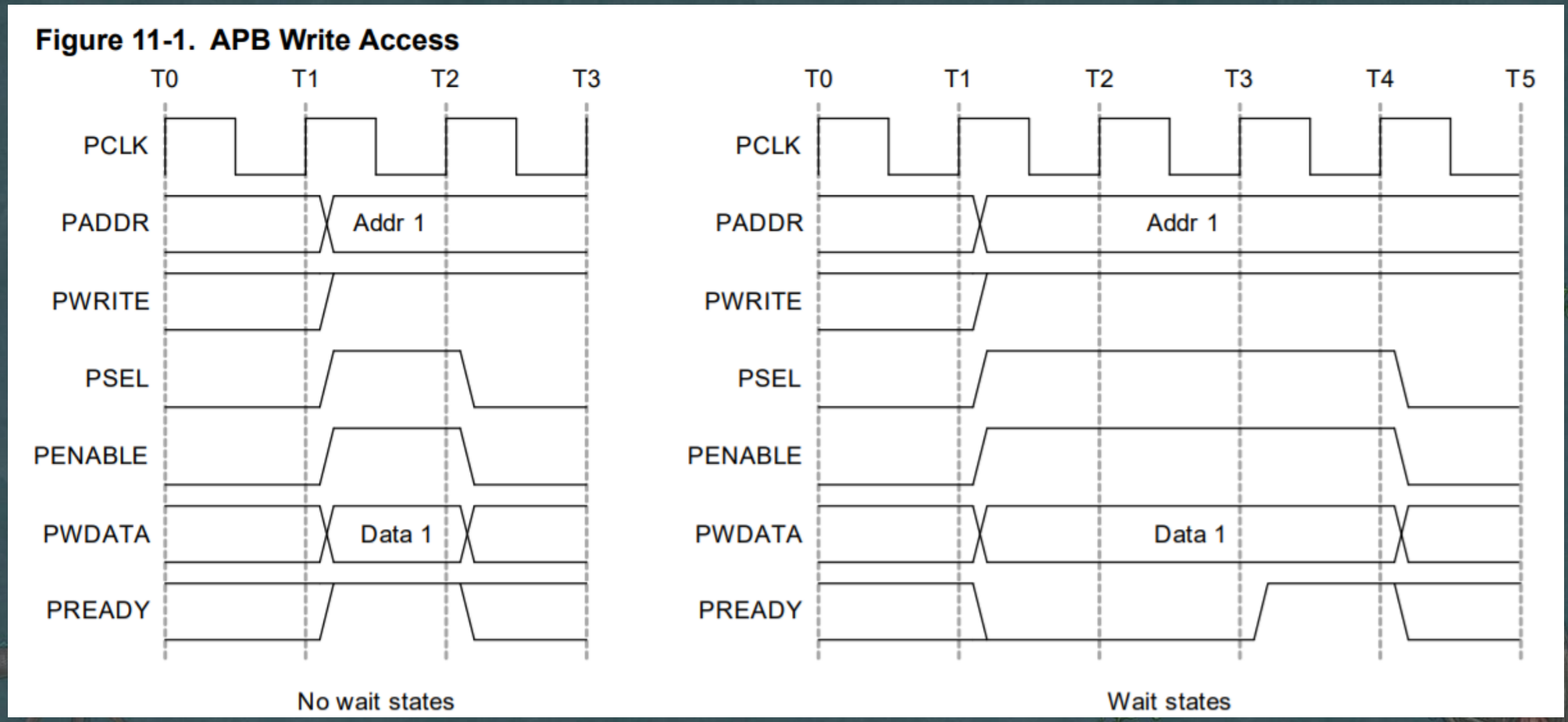
Use the Map File

Problem	Map Tool
	Look at summary
	Diff with good map file
	Find/write viewer
	Search for address nearby
	Search for variable name
Running too slow	**Statistical sampling (hard)**
	Read each and every line

Wait, who is here for the pirate jokes? Why haven't there been any pirate jokes?

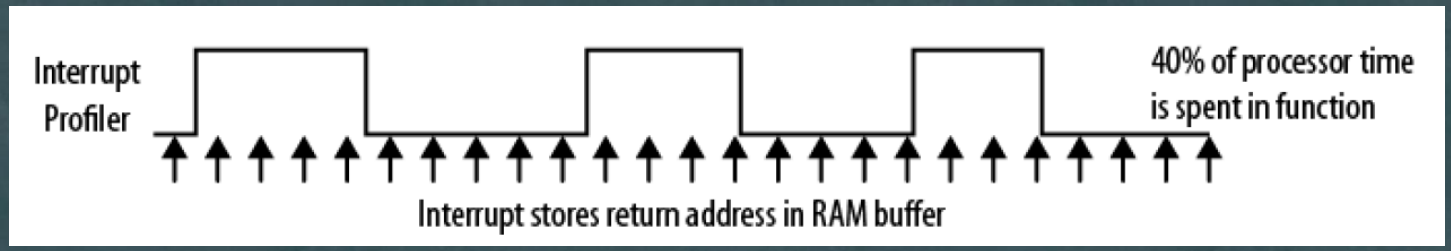
Wait State Sadness

Fast CPU and Slow Memory



Statistical Sampling Profiler

What are you doing now? What about now? Now? Now?



Use the Map File

Problem	Map Tool
Not enough RAM	Look at summary
Not enough code space	Diff with good map file
Hard fault errors	Find/write viewer
Weird memory errors	Search for address nearby
Planning FW update	Search for variable name
Running too slow	Statistical sampling (hard)
	Read each and every line

Not every tool works for every problem.

Use the Map File

Problem	Map Tool
Not enough RAM	Look at summary
Not enough code space	Diff with good map file
Hard fault errors	Find/write viewer
Weird memory errors	Search for address nearby
Planning FW update	Search for variable name
Running too slow	Statistical sampling (hard)
	~~Read each and every line~~

Some solutions are only good as soporifics.

CircuitPython on SAMD21 Map

github.com/adafruit/circuitpython
GCC generated maps are not pretty

Requires linker flags for generation:

```
LDFLAGS += -Wl,-Map=output.map,--cref
```

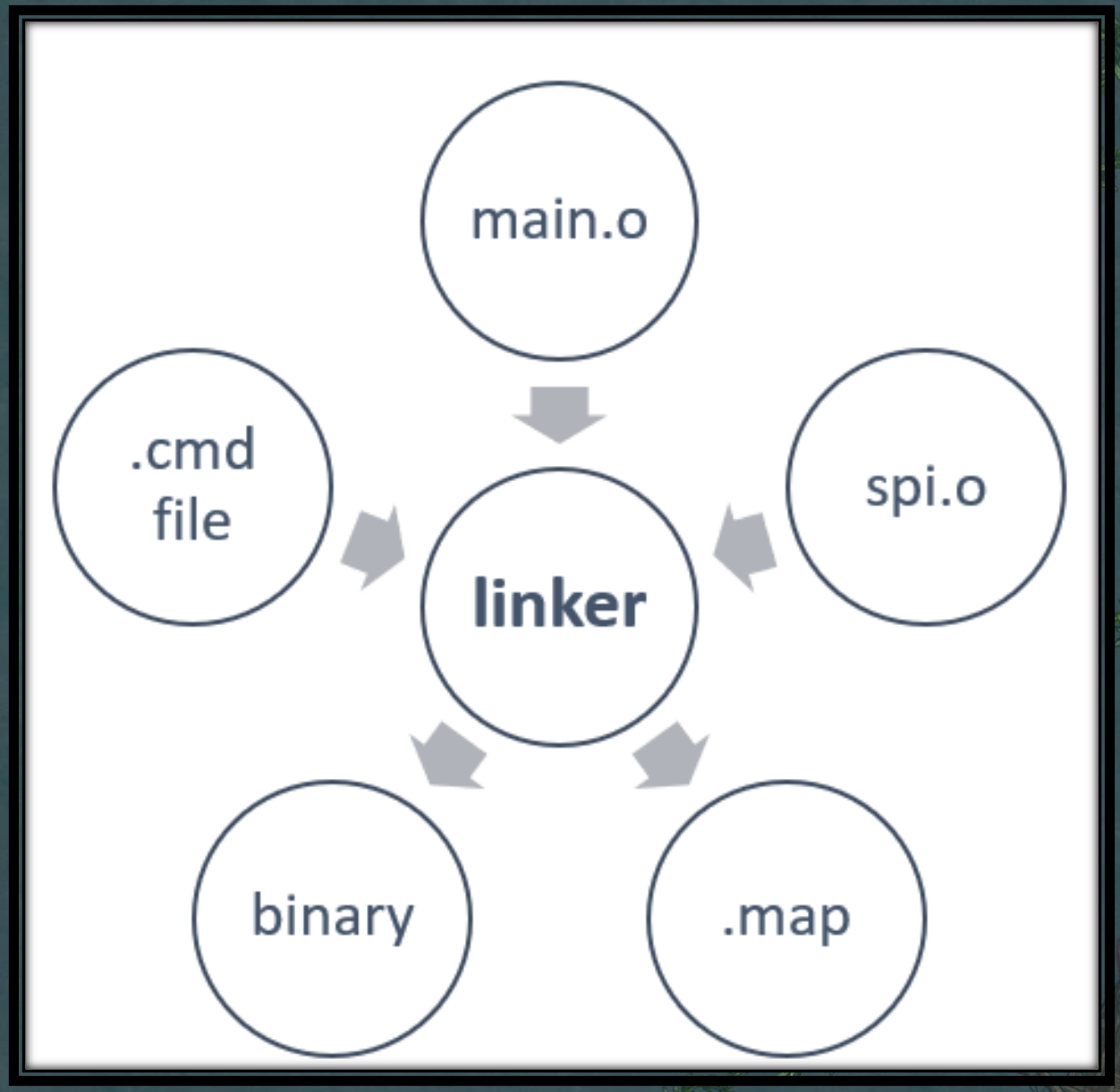
Seeduino XIAO ATSAMD21G18A-MU (ARM Cortex-M0+)

Heap

Everything else is in the heap

```
5273    .bss.yasmarang_dat
5274                    0x0000000020001930        0x1 firmware.elf.lto.o
5275    *(COMMON)
5276                    0x0000000020001934                . = ALIGN (0x4)
5277    *fill*          0x0000000020001931        0x3
5278                    0x0000000020001934                _ezero = .
5279                    0x0000000020001934                _ebss = .
5280
5281    .stack          0x0000000020001934      0xe00 load address 0x00000000000309b4
5282                    0x0000000020001934                . = ALIGN (0x4)
5283                    0x0000000020002734                . = (. + 0xe00)
5284    *fill*          0x0000000020001934      0xe00
5285                    0x0000000020002734                . = ALIGN (0x4)
5286
```

Where do map files come from?



Linker and Map

How did you get to be this way?

ld accepts Linker Command Language files written in a superset of AT&T's Link Editor Command Language syntax.

Learning and Teaching

Thank You!

Elecia White

Logical Elegance, Inc.

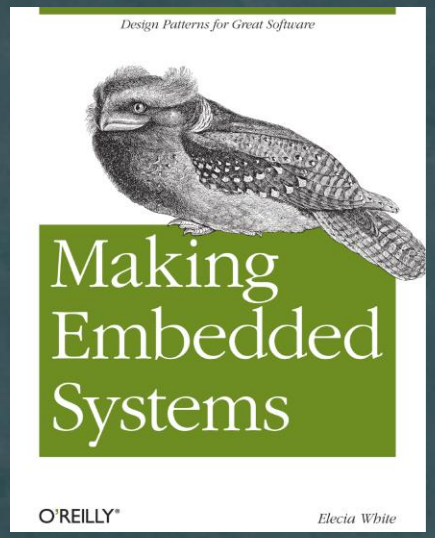
Embedded https://embedded.fm

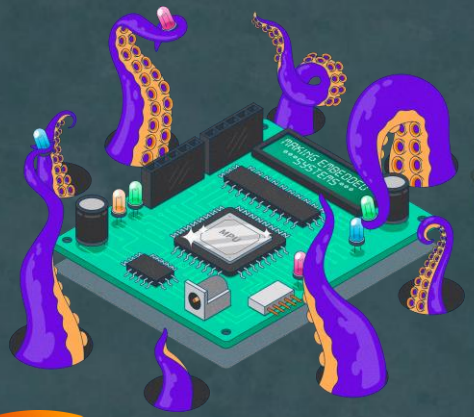
Twitter: @logicalelegance

hackaday.io/elecia

Presentation available at:

https://embedded.fm/blog/MapFiles





classpertX

EMBEDDED.fm

Acknowledgements

All mistakes are my fault, but these people helped make this presentation much better.

Christopher White

Chris Svec

Ben Hencke

Ben Hest

Keith Burzinski

Mike Szczys

Links

Explore more from these posts:

- Phillip Johnston's Linker-Generated Variables in Libc (Embedded Artistry)
- Thea Flowers' The most thoroughly commented linker script (probably)
- Cyril Fougeray's Get the Most Out of the Linker Map File (at Memfault)
- Govind Mukundan's Analyzing the Linker Map (at EmbeddedRelated)

Memory Map Land created with Inkarnate.com

ARM GCC options https://gcc.gnu.org/onlinedocs/gcc/ARM-Options.html

GNU linker (ld) options man page

Embedded.fm is at https://embedded.fm. It is also available in most podcast apps.

Elecia's book is Making Embedded Systems. Her course of the same name is through ClasspertX.

She is a co-founder of Logical Elegance, Inc.

Map Visualizers

I'm not endorsing any of these

Puncover: github.com/HBehrens/puncover

Emma: github.com/bmwcarit/Emma

amap: sikorskiy.net/prj/amap/index.html

Bloaty: github.com/google/bloaty

GccMapVisualizer: github.com/jotux/GccMapVisualizer


Thank you!
Goodbye...