

# Embedded Software Essentials

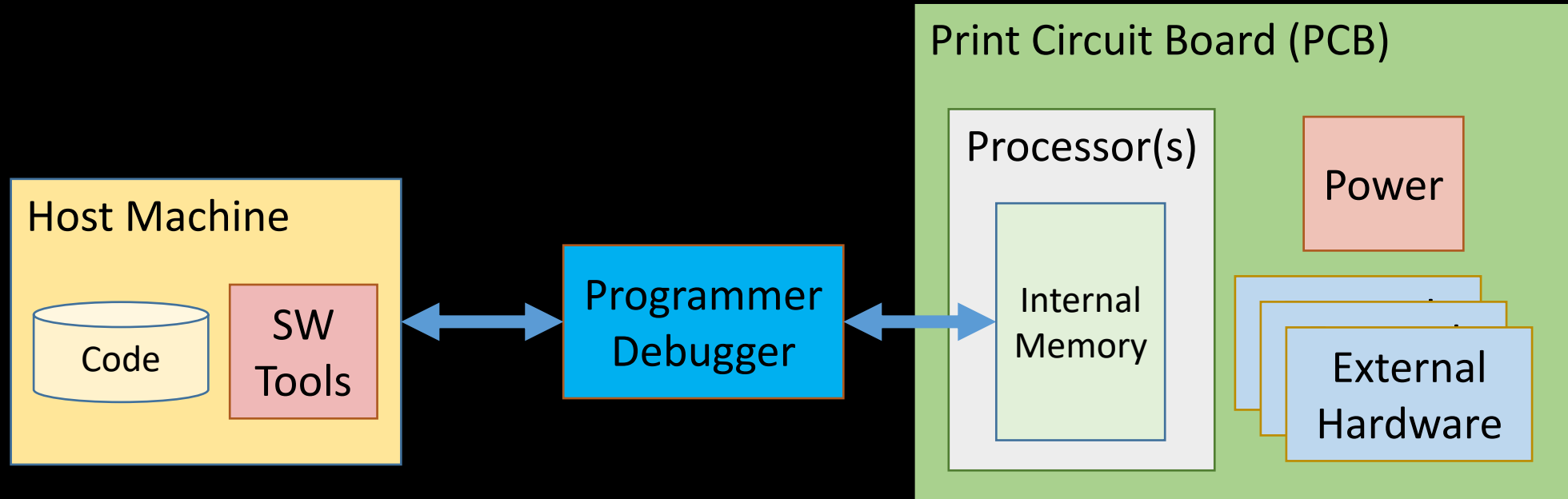
*Introduction to Build Systems using GNU Toolsets*

**C1 M2 V1**

# Copyright

- Copyright (C) 2017 by Alex Fossdick. Redistribution, modification or use of this presentation is permitted as long as the files maintain this copyright. Users are permitted to modify this and use it to learn about the field of embedded software. Alex Fossdick and the University of Colorado are not liable for any misuse of this material.

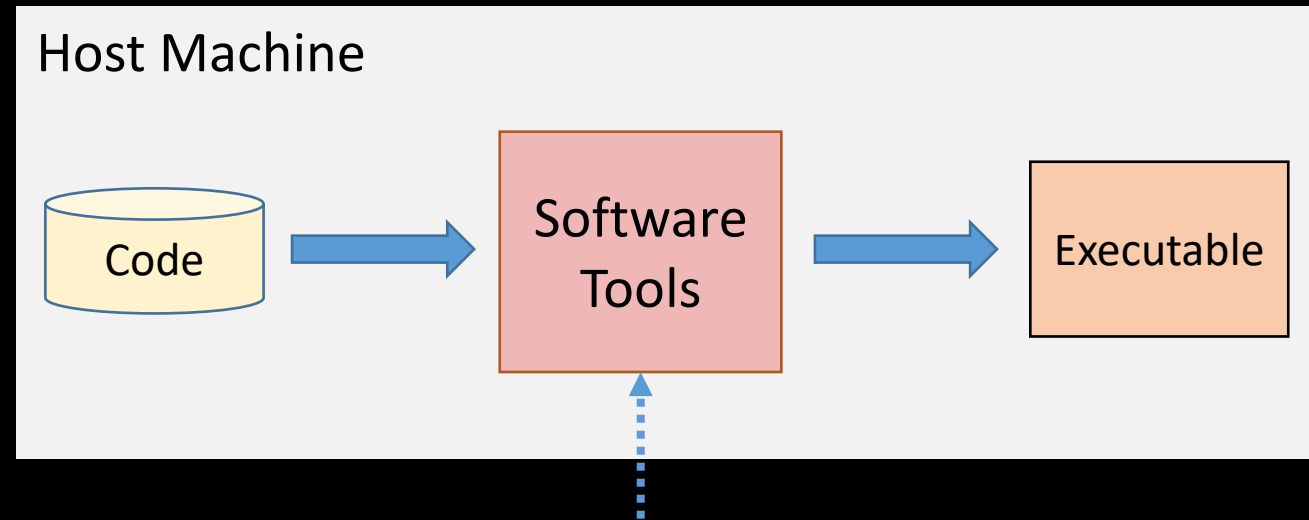
# Embedded System Development Platform



**The host machine contains our  
Build Environment**

# Build Environment

The host machine contains our Build Environment

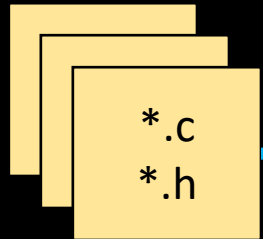


**Software Engineer's Tools include Compiler Toolchain**

- GCC – GNU's Compiler Collection
- Make

# Software Tools

Source Files usually  
mostly in High Level  
Languages



## Compiler Toolchain

Software Tools

Compiler  
Toolchain

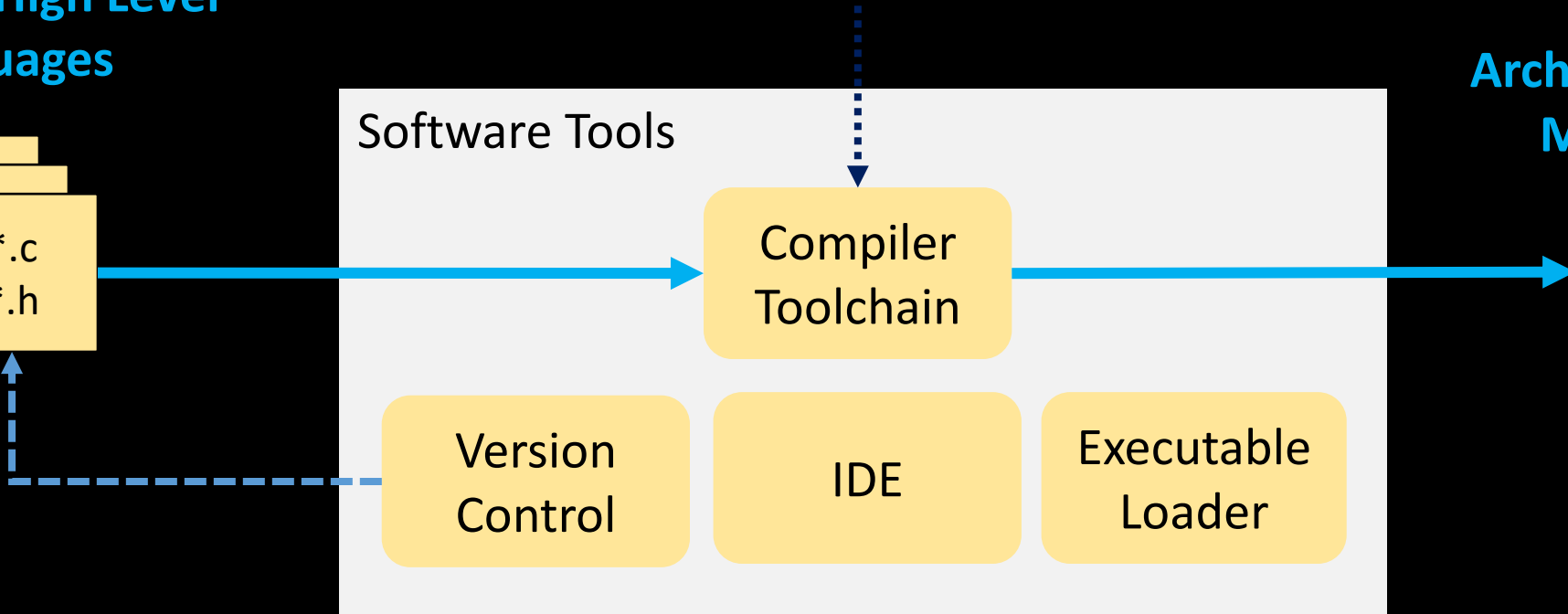
Version  
Control

IDE

Executable  
Loader

Architecture Specific  
Machine Code

Executable  
File



# Building a Software Project

## C-Programming (High Level Language)

```
int x = 0;
int y = 20;
int z = 5;

...
while (y >= z) {
    y = y - z;
    x++;
}
```

## ARM Assembly Language (Low Level Language)<sup>[1]</sup>

```
ldr    r2, (y)
ldr    r3, (z)
ldr    r4, (x)

LOOP:
sub    r2, r3
inc    r4
cmp    r2, r3
bgt    LOOP
str    r2, (y)
str    r4, (x)
```

[1] (x),(y),(z) = Pseudocode

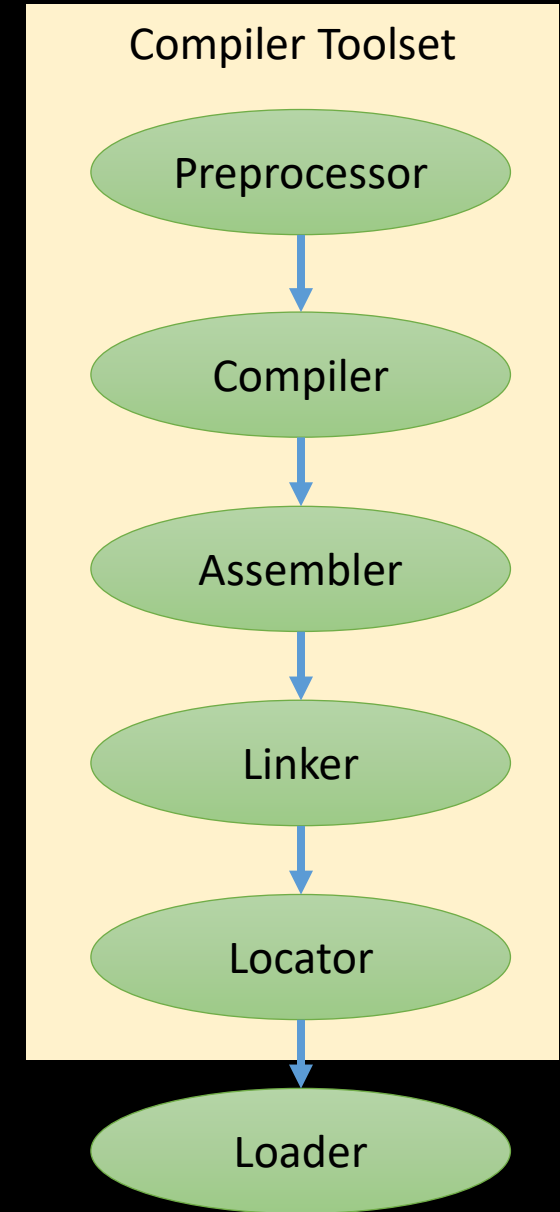
## Machine Code (Binary encoded Assembly Instructions) <sup>[2]</sup>

```
0x0c1b
0x7023
0x2302
0x71bb
0x2300
0xf7ff ef24
0xc407
0x8023
0x3402
```

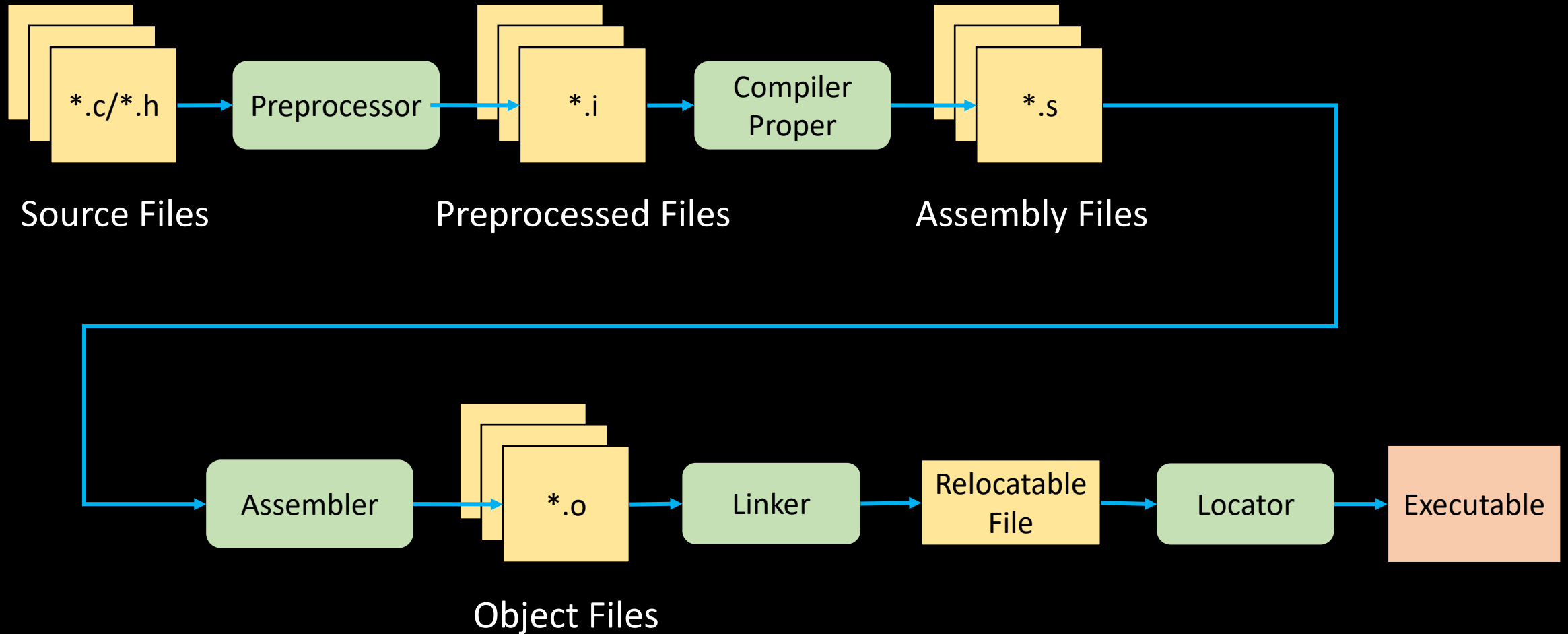
[2] Machine code just an example

# Building a Software Project

- Build and Install Process:
  - Preprocessing
  - Assembling
  - Compiling
  - Linking
  - Locating
  - Installing
- Installation will require other tools

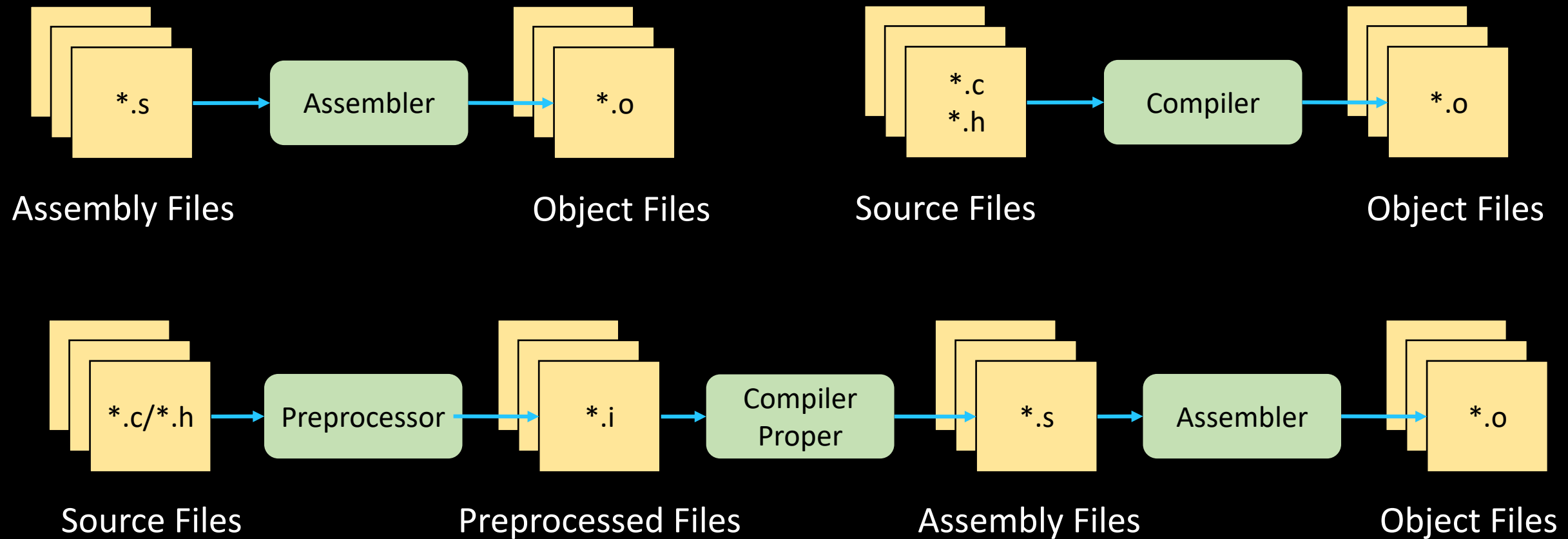


# Build Process (linear)

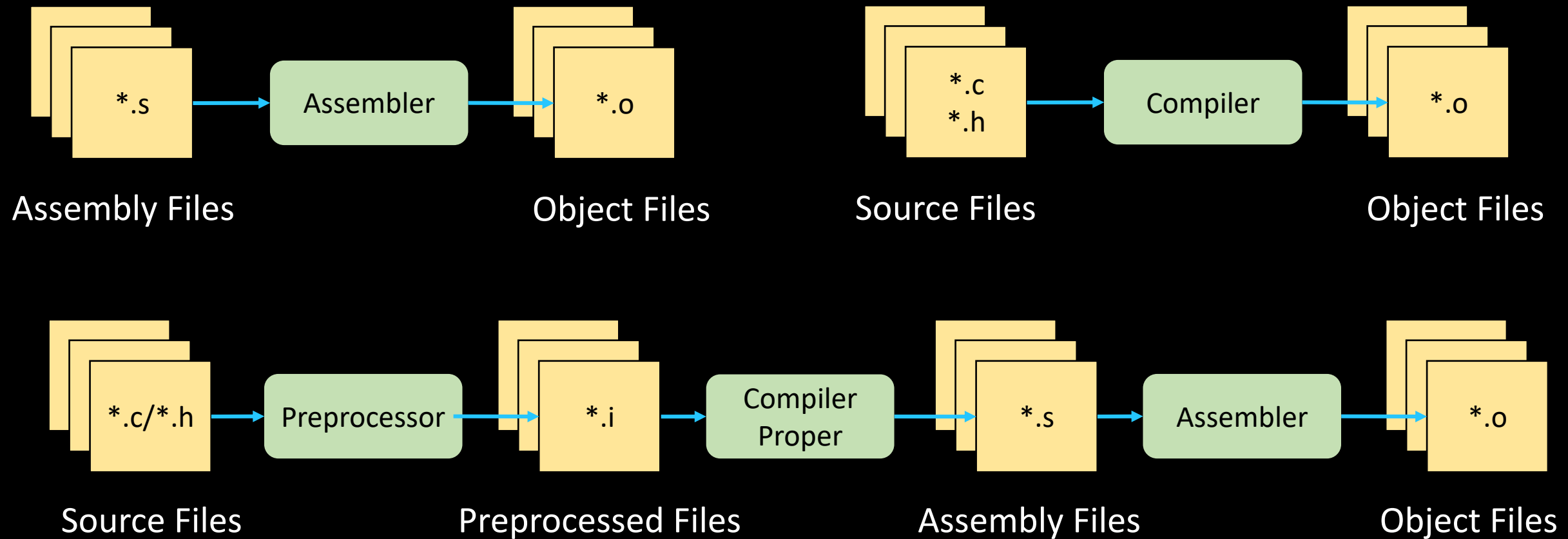




# Compilation (No Linking)

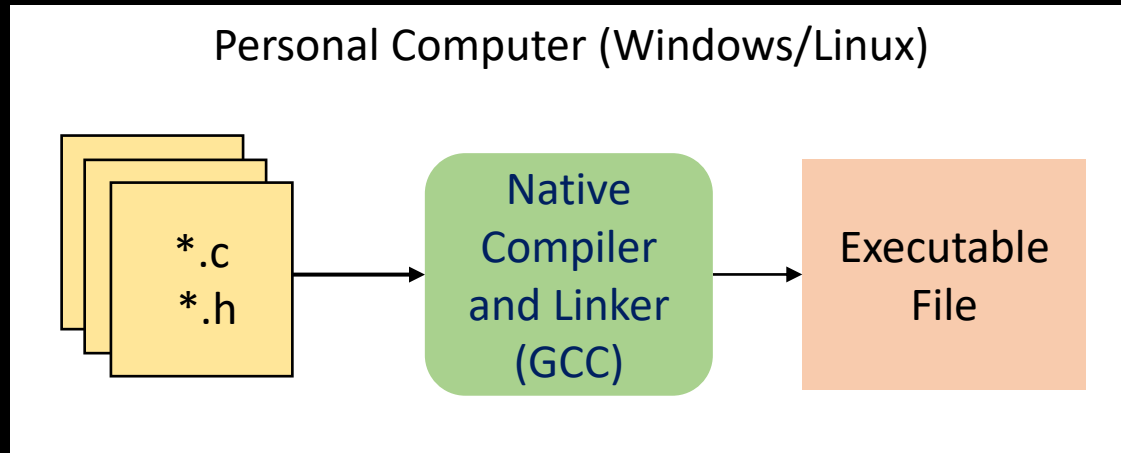


# Compilation (No Linking)



# Native Compilation

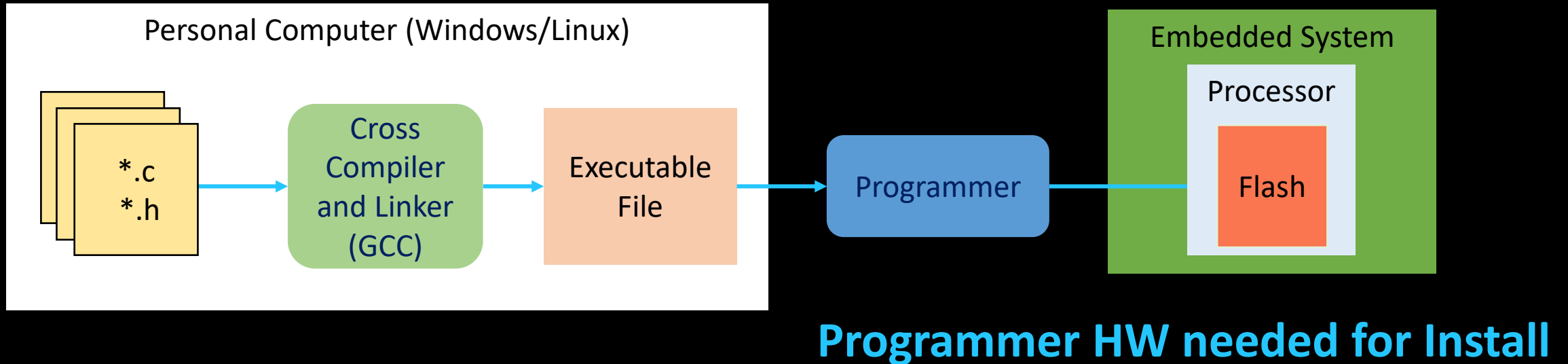
- Compile an executable on one system and it is intended to run on same system



**No hardware needed**

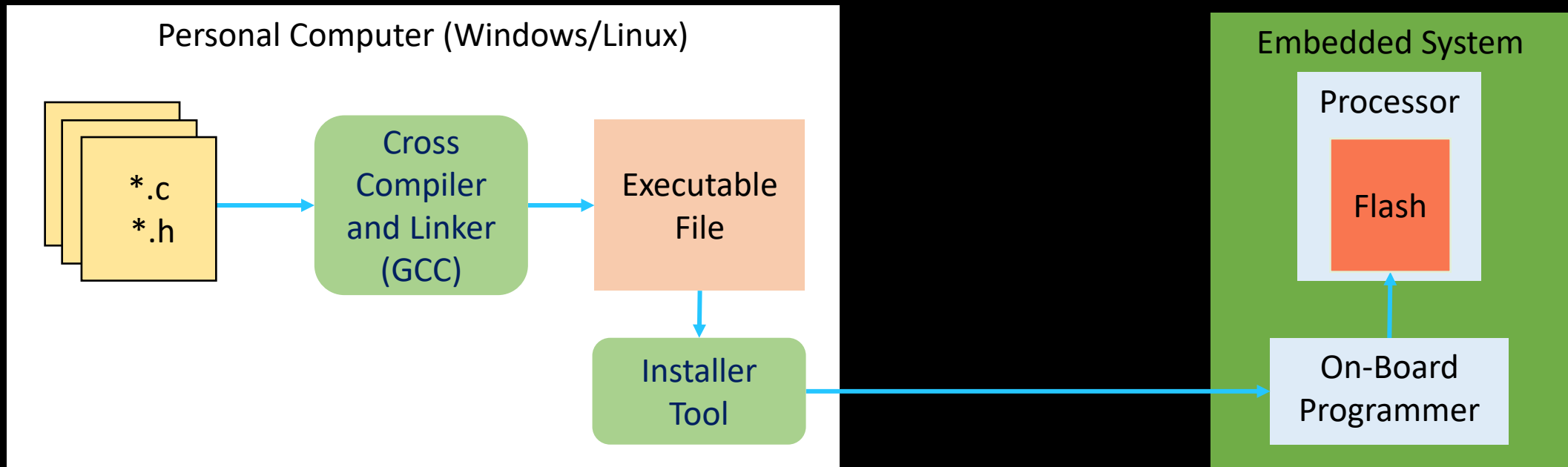
# Cross Compilation

- Compile an executable on one system and it is intended to run on another



# Cross Compilation

- Installer tool sends executable to on board programmer
  - No external hardware needed



# Compiler Toolchain

- **GCC = GNU's Compiler Collection**
  - Contains many tools (compiler, assembler, linker, etc)
- **GNU Make**
  - “Tool that controls the generation of executables and other non-source files of a program from the program's source files”<sup>[2]</sup>

