Game Boy Development

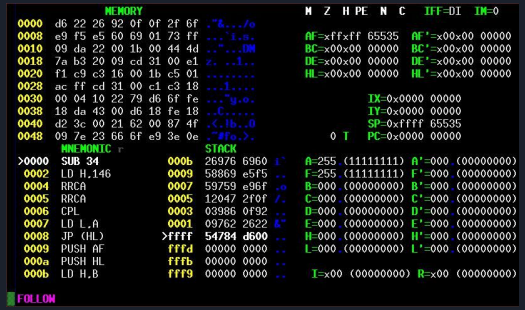
Led by Will McGloughlin, Noah Stansbury, Luke Miller, and Cole Baughman

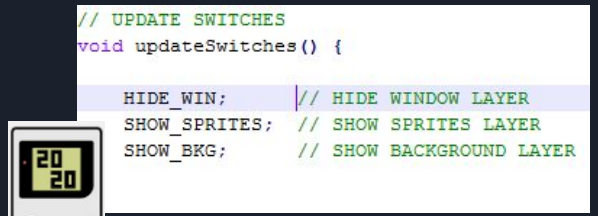
The Nintendo Game Boy

- 8-bit handheld game console released in 1989
- Display
  - 160x144 dot matrix
  - 2-bit color (4 shades)
- CPU
  - Sharp LR35902 (GBZ80)
  - Based on 8080 and Z80 processors of the time
  - Used a version of Assembly
    - Based on Z80 assembler
- Memory
  - 8KiB RAM built in
    - More might be in carts
  - 8KiB RAM just for display
  - 32KiB ROM in cart for game code
- Misc
  - Graphical units called sprites

Programming

- Programmers of the time would have had to learn GBZ80 asm
- The Game Boy Development Kit (GBDK)
  - Rocky development
    - Development started late 90s
    - Abandoned early 2001
    - Rebooted in 2020
  - Compiles C into asm that the Small Device C Compiler (SDCC) can understand
  - In short, you can make Game Boy games with C!
- Your code
  - You'll write code in **main.c**
  - Everytime you make changes, run **make.bat**
    - This will turn your code into a .gb rom file
    - Make sure to run make in the same folder as the gbdk
    - Run the .gb file with VisualBoyAdvance





C Background

- C is a lower level programing language compared to the languages you may know like Python or Java
  - Python allows us to be more flexible, for example variable types can be determined at runtime by the data assigned to them
  - C code lacks this flexibility
- C must be compiled before it can be run
  - In this workshop we have given you the file **make.bat** it will compile your C code into .gb files for the emulator
- We are going to go over some syntax you'll need to know for your development

C Syntax - Semicolons and Variable Types

- For this workshop we will be using statements similar in nature to ones you know from other programing languages
- Syntax of those statements will be different
- For example:
  - Every non blocking statement must end with a semicolon ;
    - This would not include statements like if or while, the are followed by brackets {} containing the statements to be executed by them
  - Variables must be declared with a type
    - int x;
    - int y = 4;
    - float i = 3.4;
- Print using printf(“Text to be printed”);

```
void main(void)
{
    printf("Hello World");

    // Loop forever
    while(1) {


                // Game main loop processing goes

                // Done processing, yield CPU and
        wait_vbl_done();
    }
}
```

C Syntax - Main Function and Other Functions

- Similar to Java, C runs code that is inside the main function
  - From here you can call other functions
  - They may be in the same .c file or other .c files
- In C it is customary to have header files that outline functions used in the program
  - Implement functions in a .c file of the same name
- Include these header files and other files in main.c with include statements at the top of the program like **#include “block.h”**

C Syntax - If-else Statements

- For this workshop need to know if statements
  - Similar syntax to Java
- Possible conditional operators used in this workshop include
  - `==` equals
  - `!=` not equal
  - `&&` and
  - `||` or
  - `>` greater than
  - `<` less than

```
if (test expression) {
    // run code if test expression is true
}
else {
    // run code if test expression is false
}
```

C Syntax - Calling Functions

- Calling functions is similar to Java
  - Functions for this workshop are created by GBTD
- We will need to call them in main
- Also, we will need to supply them with arguments if the function requires them
- For example a function with two arguments
  - function_name(argument1, argument2);
- Calling a function with no arguments leaves the parentheses blank
  - function_name();

```
int sum(int a, int b)
{
     int c=a+b;
     return c;
}

int main(
{
    int var1 =10;
    int var2 = 20;
    int var3 = sum(var1, var2);
    printf("%d", var3);

    return 0;
}
```

Download workshop materials

1. Navigate to https://github.com/psuieee/projects2021/releases
2. Download workshop_materials.zip
3. Extract the contents to an easy-to-find place

Hello, World!

A word on GBTD

Getting your sprite on screen

Getting that sprite to move

Final changes

Next Steps

- Background tiles
- Multiple Sprites
- Larger sprites
- Sprite groups
- Sound

See GBDK documentation!