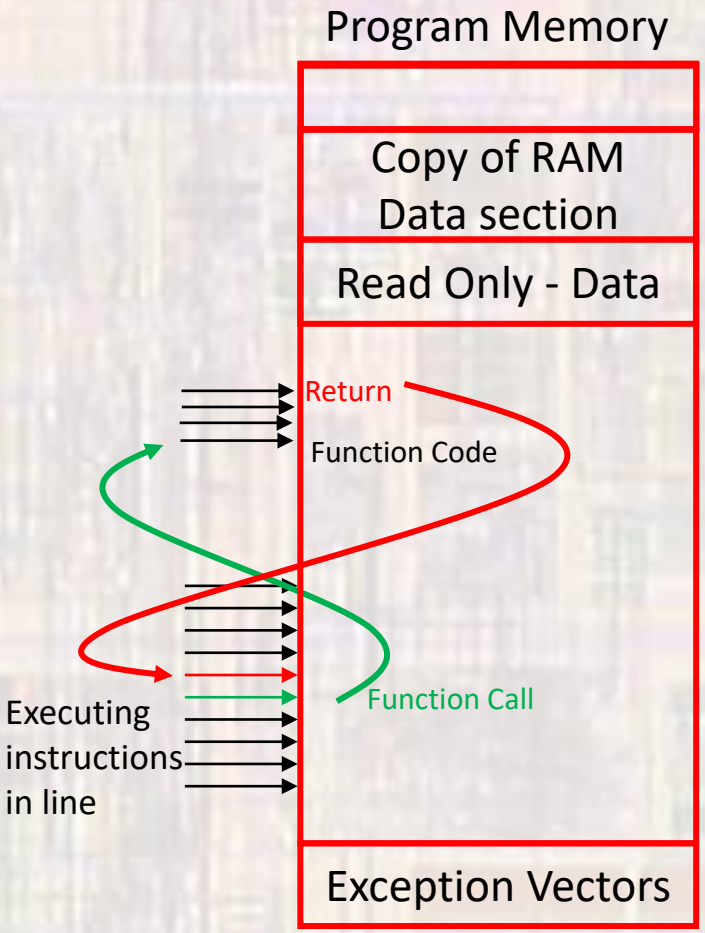
Non-Linear Program Execution - Functions

Last updated 6/16/23

These slides show program execution flow using functions

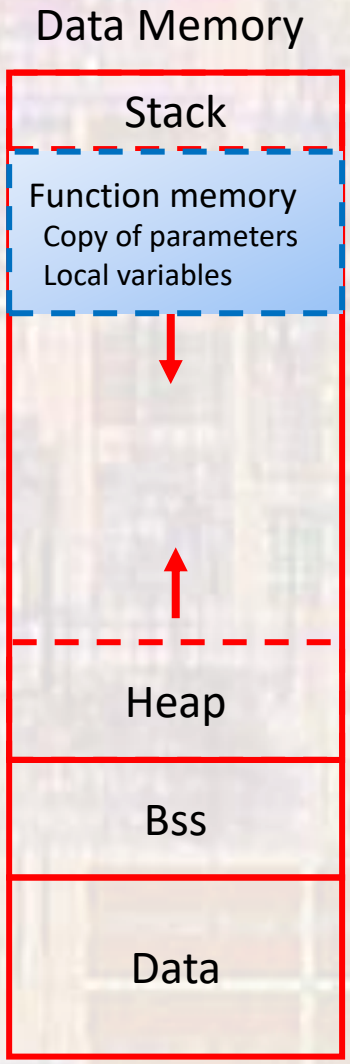
Non-Linear Program Execution - Functions

- Functions and Memory - program memory
  - Function call transfers execution to a separate section of code (function)
  - When done, the function returns to the next line of the calling function code
  - Multiple calls to the same function transfer execution to the same location
    - Only one copy of the function in program memory



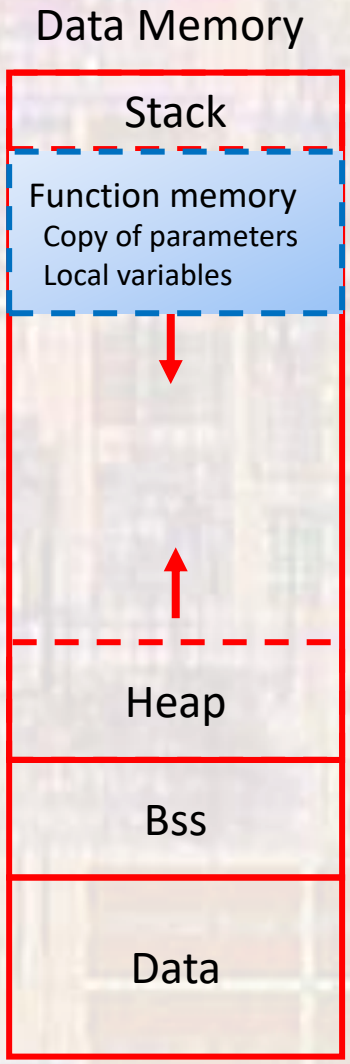
Non-Linear Program Execution - Functions

- Functions and Memory – data memory
  - Function call creates a space in the stack
    - Called a Stack Frame
  - Function operates in this newly created space (scope)
  - When the function returns, the space is reclaimed (not necessarily erased but no longer available)



Non-Linear Program Execution - Functions

- Functions and Memory – data memory – Stack Frame
  - Storage order – system dependent
    - Arguments passed to the function
      - In the order they are declared in the function call.
    - Return address
      - The address of the next instruction after the function call
    - Frame pointer
      - Pointer to the current stack frame
      - Can be stored on the stack or in a special register
    - Local function variables
  - Removal order
    - Reverse of the storage order
    - The return value can be stored on the stack or in a special register in the processor



Non-Linear Program Execution - Functions

- User Defined Functions – Data memory

declaration

```
float ave(float val1, float val2);
…

int main(void){
  float average;
  float try1;
  float try2;

  // enter try1, try2
  …
  average = ave(try1, try2);
  …
  return 0;
}
```

call

definition

```
float ave(float val1, float val2){
  float tmp_val;
  tmp_val = (val1 + val2)/2;
  return tmp_val;
}
```