ITS553 – Secure Java Programming  
Instructor: Dr. Michael Tu  
Homework Assignment #1  
Author: Steve Jia  
Date: 2017-JUN-21  
Description: explanation of sample program running on a computer

**Homework 1: Sample Program**

1. *int x =100;*
2. *int main(){*
3. *int x =0, y, z;*
4. *x= x+1;*
5. *y =2;*
6. *z = x+y;*
7. *return z;*
8. *}*

**Homework 1: Explanation**

When the program runs, the computer first loads the program from the computer’s I/O device, usually the hard drive, into main memory. In the diagram, the code is stored in the “code” section. From there, the CPU will take in instructions from the application’s memory space and process them line by line. First, the CPU receives the first instruction and allocates a section in the main memory for the main stack frame to be created. A base address will be returned to the CPU via the address bus.

For the first line of code, int x=100, the CPU assigns a value of 100 first to register zero, obtains the address of the global variable x using the address bus, and then uses a store command to save the value 100 from register zero to the memory address of the global x in the static storage in the main memory using the address bus and the data bus. In the main stack frame, the static link section will be pointed to the global variable x in the static storage [Figure 1].

For line 3, the CPU first loads the value of zero to register zero, obtains the address of the local variable x, and then uses the address bus and data bus to store the register zero value to the memory address of x. For local variables y and z, the CPU will do the same thing and initializes them with values of zero. In the main stack frame, memory addresses of x, y, and z will appear in the local variables section, all with values of zeroes [Figure 2].

For line 4, the instruction and data will come to the CPU via the data bus and the address of local variable x will come to the CPU via the address bus. A value of x will be loaded into register 0 and a value of 1 will be loaded into register 1. Once the control unit in the CPU gives the go-ahead for the “x=x+1” instruction, the ALU will receive and add the values in register 0 and register 1 together, and stores the result in register 0. After this, the summation result will be stored in local x’s memory address via the data bus. The value of local x is updated in the main stack frame [Figure 3].

Line 5 is an assignment for local variable y. The CPU loads a value of 2 into register 1, and then stores the value into variable y via the data bus after obtaining the memory address via the address bus. The value of y is updated in the main stack frame [Figure 4].

Memory addresses of x, y, and z are send to the CPU via the address bus in line 6. Instructions and data are loaded into the CPU via the data bus, and the CPU loads the value of x into register 0 and the value of y into register 1. The control unit will then send the “z=x+y” instruction and the register data to the ALU via the control bus, and the ALU will perform the addition and restores the result in register 0. After the calculation, the result will be stored in z’s memory address via the data bus back in the main stack frame [Figure 5].

Line 7 of the program will return the value of z, marking the end of the program. The CPU will receive the return address and z’s address from main memory, load z’s value into register 1 by reading z’s memory address, and then moves the value in register 1 to the return register %i0.

Line 8 will generate the nop assembly command, finishing the program.

**Homework 1 – Stack Frame Figures**

Figure - Main Stack Frame After Program Starts

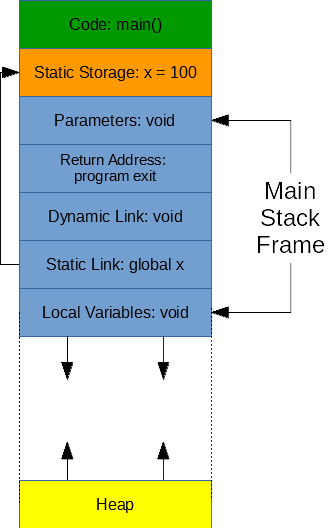


Figure - Main Stack Frame After Local Variable Initialization

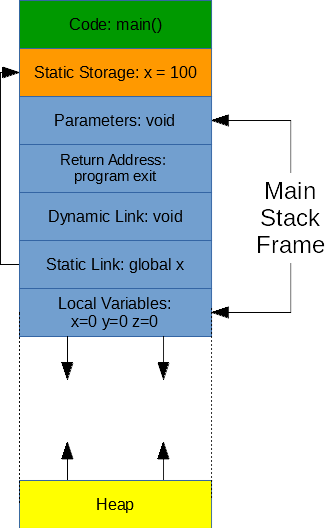


Figure - Main Stack Frame After x=x+1

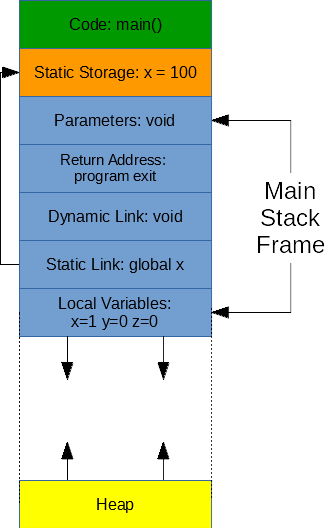


Figure - Main Stack Frame After y=2

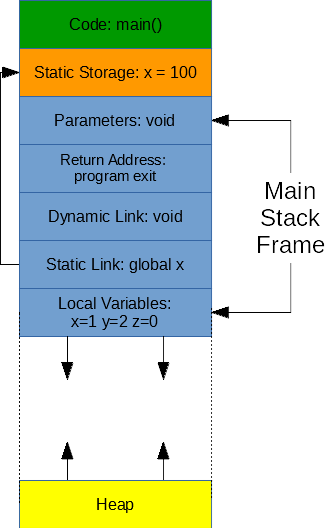


Figure - Main Stack Frame After z=x+y

