ECE 220 Computer Systems & Programming

Lecture 10 – Run-Time Stack

October 3, 2017

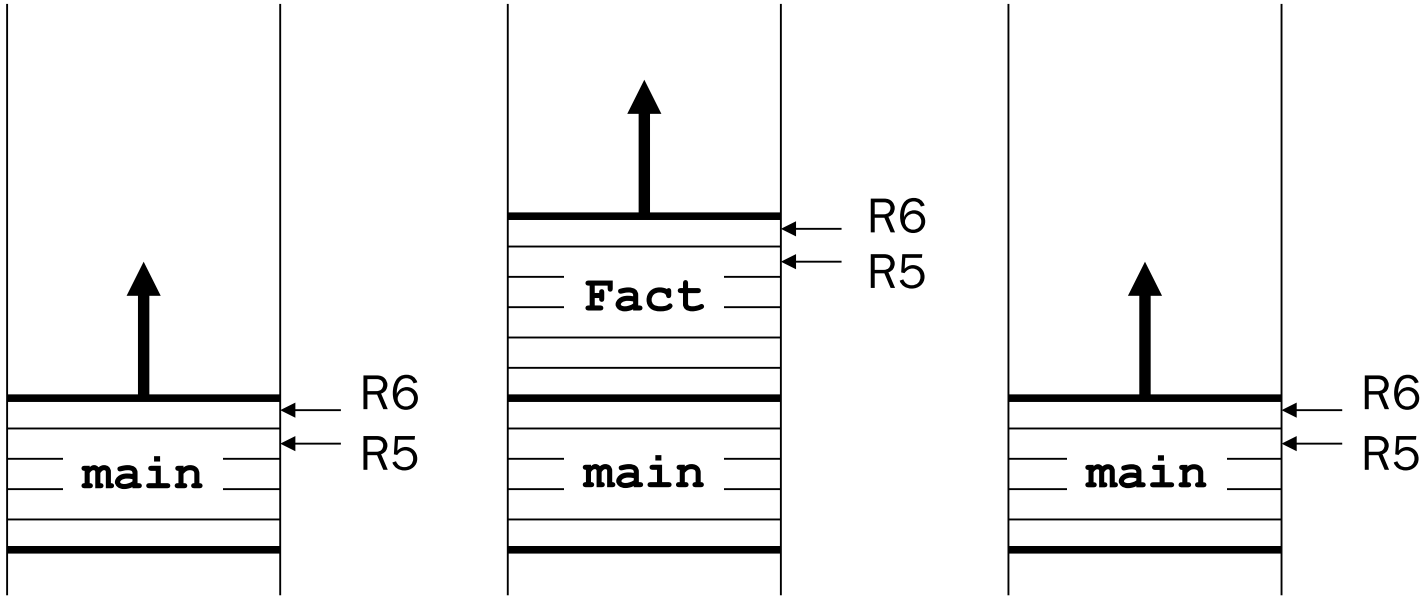
- MP4 due Thursday, 10/5, at 10pm
- Quiz 2 next week – reserve your seat soon

ECE ILLINOIS

ILLINOIS

Run Time Stack

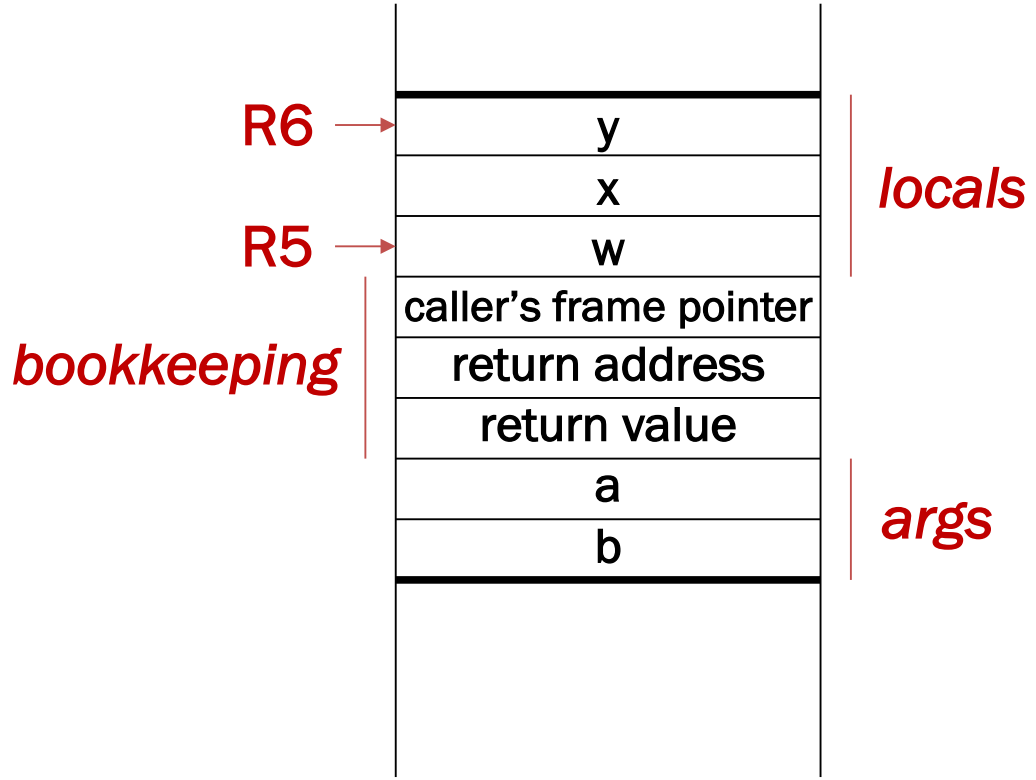
- R5 – **Frame Pointer**. It points to the beginning of a region of activation record that stores local variables for the current function.
- R6 – **Stack Pointer**. It points to the top most occupied location on the stack.
- Arguments are pushed to the stack from **right to left**.
- Local variables are pushed to the stack **in the order they are declared**.



Before call *During call* *After call*

Activation Record

```
int func(int a, int b)
{
   int w, x, y;
   .
   .
   .
   return y;
}
```



Stack Built-up and Tear-down

Caller function

1. *caller setup*: push callee's arguments onto stack
2. pass control to callee (invoke function)

Callee function

3. *callee setup*: push bookkeeping info and local variables onto stack
4. execute function
5. *callee teardown*: pop local variables, caller's frame pointer, and return address from stack
6. return to caller

Caller function

7. *caller teardown*: pop callee's return value and arguments from stack

Run-Time Stack Example

The call: **w = Volta(w, 10);**

;Watt program

1. **Caller setup**

   (push callee's arguments onto the run-time stack)

   ```
   ; push 10
   ```

   ```
   ; push w
   ```

2. **Caller pass control to callee**

```
int main(){
        int a;
        int b;
        …
        b = Watt(a);
        b = Volta(a,b);
        return 0;
}

int Watt(int a){
        int w;
        …
        w = Volta(w, 10);
        return w;
}

int Volta(int q, int r){
        int k;
        int m;
        …
        return k;
}
```

Starting the Callee Function

Volta

3. Callee setup

(push bookkeeping info & local variables onto the run-time stack)

```
; leave space for return value

; push return address (R7)


; push caller's frame pointer (R5)


; set new frame pointer

; push local variables
```

4. Execute function

(implement function logic)

…

Ending the Callee Function: **return k;**

5. **Callee teardown**

   (pop local variables, O.F.P., and return address from the run-time stack)

```
; copy k into return value


; pop local variables

; pop caller's frame pointer (into R5)


; pop return addr (into R7)
```

6. **Return to caller**

   (R6 should be pointing to return value at this moment)

Returning to the Caller Function: **w = Volta(w,10);**

```
Code after 'JSR Volta'
;Watt program
```

7. Caller teardown

(pop callee's return value and arguments from the run-time stack)

```
; load return value at top of stack (R6)

; perform assignment (w = Volta(w,10))

; pop return value

; pop arguments
```

See Page 394 on the textbook for putting all these LC-3 code together

Run-Time Stack Exercise

```
#include <stdio.h>
int Fact(int n);

/* main function */
int main() {
    int number;
    int answer;

    printf("Enter a number: ");
    scanf("%d", &number);
    answer = Fact(number);
    printf("factorial of %d is %d\n", number, answer);
    return 0;
}
```

```
/* Function definition of Factorial function */
int Fact(int n) {
    int i, result=1;

    for (i = 1; i <= n; i++)
        result = result * i;

    return result;
}
```