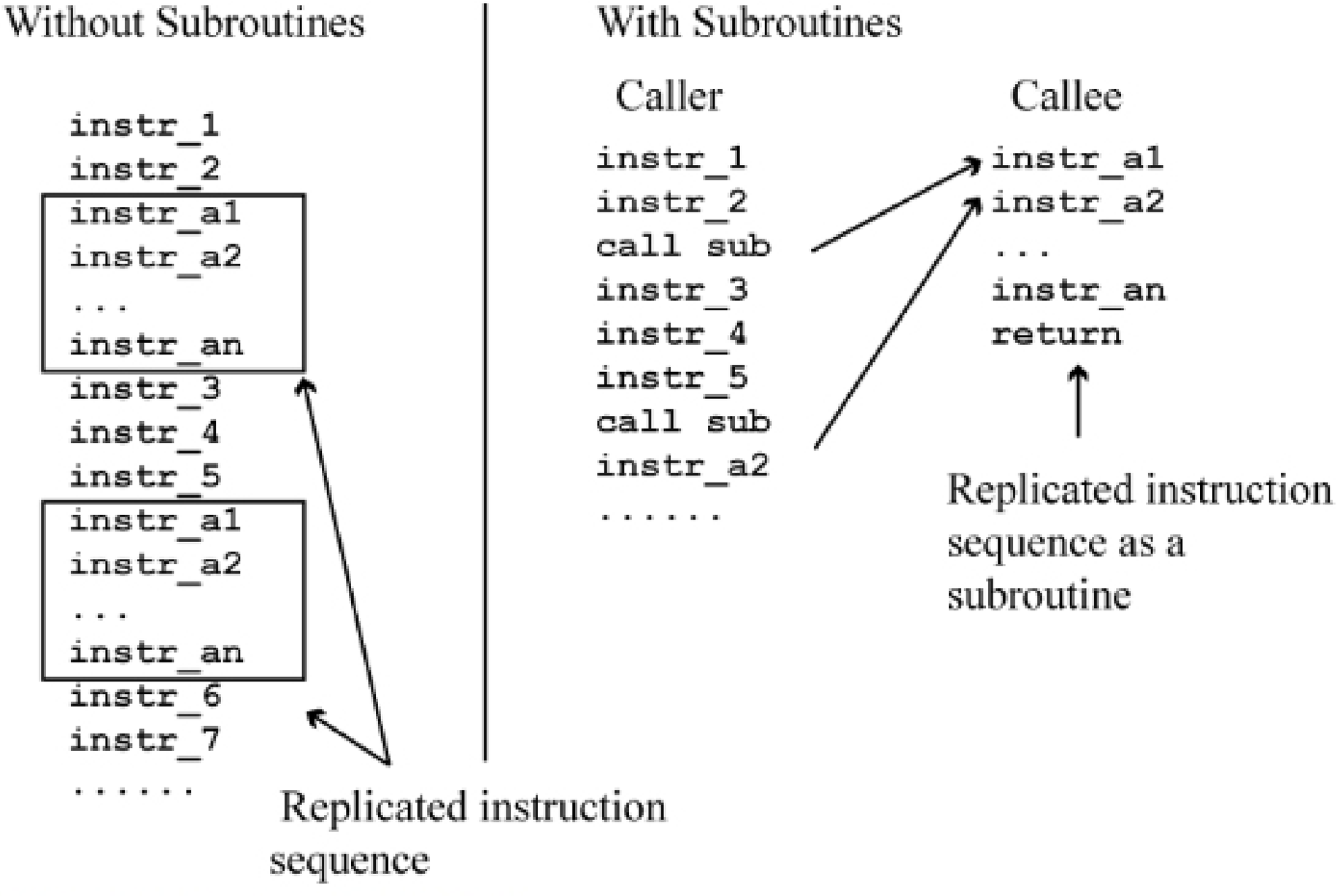
Subroutines

- A **subroutine** is a block of code that is **called** from different places from within a main program or other subroutines.
  - Saves code space in that the subroutine code does not have to be repeated in the program areas that need it; only the code for the *subroutine call* is repeated.
- A subroutine can have zero or more **parameters** that control its operation
- A subroutine may need to use **local variables** for computation.
- A subroutine may pass a **return value** back to the caller.
- Space in data memory must be reserved for parameters, local variables, and the return value.

Why Subroutines?



C Subroutines (aka. Function)

General form of a *C* subroutine is:

```
(return_type) subname (parm list)
{
  local_variable_decl;
  subroutine_body;
  return(return_value);
}
```

The statement i << j is valid in *C*, this is only used for example purposes.

`vlshift` Subroutine

```
 // variable left shift
unsigned char vlshift(unsigned char v,
unsigned char amt)
{
  while (amt) {
    v = v << 1;
    amt--;
  }
  return(v);
}

main(void){
 unsigned char i,j,k;

 i=0x24; j = 2;
 k = vlshift(i,j);
 printf(
  "i=0x%x, shift amount: %d,result: 0x%x\n",
  i,j,k);
}
```

parameter list: gives types and names

subroutine body

subroutine return

main program

subroutine call