Disassembling a small C code

1) The code example :

```
#include <stdio.h>

void f(int a, int b) {
   int x, y ;
   char buf[10] ;

   x=a ; y=b ;
   while (x < y) {
        buf[x] = 42 ;
        x = x+1 ;
   }
}

int main() {
   f(1,8) ;
   return 0 ;
}
```

How to guess the **stack layout** of function `f()` …

2) Compiling

To get an x86 (32 bits) executable called `example`, without stack protections :

```
gcc -m32 -fno-stack-protector -o example example.c
```

Alternatively you can also get a binary **with debug information** :

```
gcc -m32 -fno-stack-protector -g -o example example.c
```

2) Disassembling with objdump (and look at the code)

```
objdump -S example
```

The code of function `f()` with debug information :

```
void f(int a, int b) {
    1125: 55                    push   %rbp
    1126: 48 89 e5              mov    %rsp,%rbp
    1129: 89 7d dc              mov    %edi,-0x24(%rbp)
    112c: 89 75 d8              mov    %esi,-0x28(%rbp)
   int x, y ;
   char buf[10] ;

   x=a ; y=b ;
    112f: 8b 45 dc              mov    -0x24(%rbp),%eax
    1132: 89 45 fc              mov    %eax,-0x4(%rbp)
    1135: 8b 45 d8              mov    -0x28(%rbp),%eax
    1138: 89 45 f8              mov    %eax,-0x8(%rbp)
   while (x < y) {
    113b: eb 0e                 jmp    114b <f+0x26>
        buf[x] = 42 ;
    113d: 8b 45 fc              mov    -0x4(%rbp),%eax
    1140: 48 98                 cltq
    1142: c6 44 05 ee 2a                movb   $0x2a,-0x12(%rbp,%rax,1)
```

```
     x = x+1 ;
  1147: 83 45 fc 01        addl   $0x1,-0x4(%rbp)
 while (x < y) {
  114b: 8b 45 fc           mov    -0x4(%rbp),%eax
  114e:  3b 45 f8          cmp    -0x8(%rbp),%eax
  1151: 7c ea              jl     113d <f+0x18>
 }
}
  1153: 90                 nop
  1154: 5d                 pop    %rbp
  1155: c3                 retq
```

We can see the addresses of x, y and buf (relatively to ebp) :

@x = rbp-4   `[x = x+1          [addl   $0x1,-0x4(%rbp)]`

@y = rbp-8 `[while (x < y) [cmp    -0x8(%rbp),%eax]`

@buf = rbp-20   [`buf[x] = 42 ;  movb   $0x2a,-0x20(%rbp,%rax,1)]`

3) Disassembling with IDA Pro

Using IDA Pro you can retrieve (more easily?) the same information

ida64 example

View the flow-chart of function `f()`

Rk : you can also use the web site https://godbolt.org/ to produce assembly code wrt various compiler/architecture /options ...

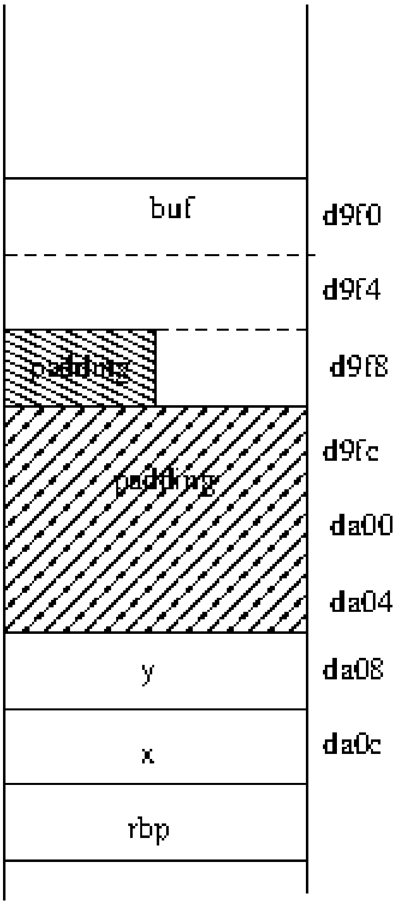
4) Debugging with `gcc`

Finally, you can also run your program under the `gcc` debuger, and print the actual addresses (at runtime) of the `f()` local variables :

```
gdb example
break f              // set a breakpoint at beginning of function f()
run                  // execution stops when startinf f()
print &x            // @x = 0x7fffffffda0c
print &y           // @y = 0x7fffffffda08
print &buf         // @buf = 0x7fffffffd9f0
```

5) Stack layout

(see next page)


buf
d9f0
d9f4
padding
d9f8
d9fc
padding
da00
da04
y
da08
x
da0c
rbp