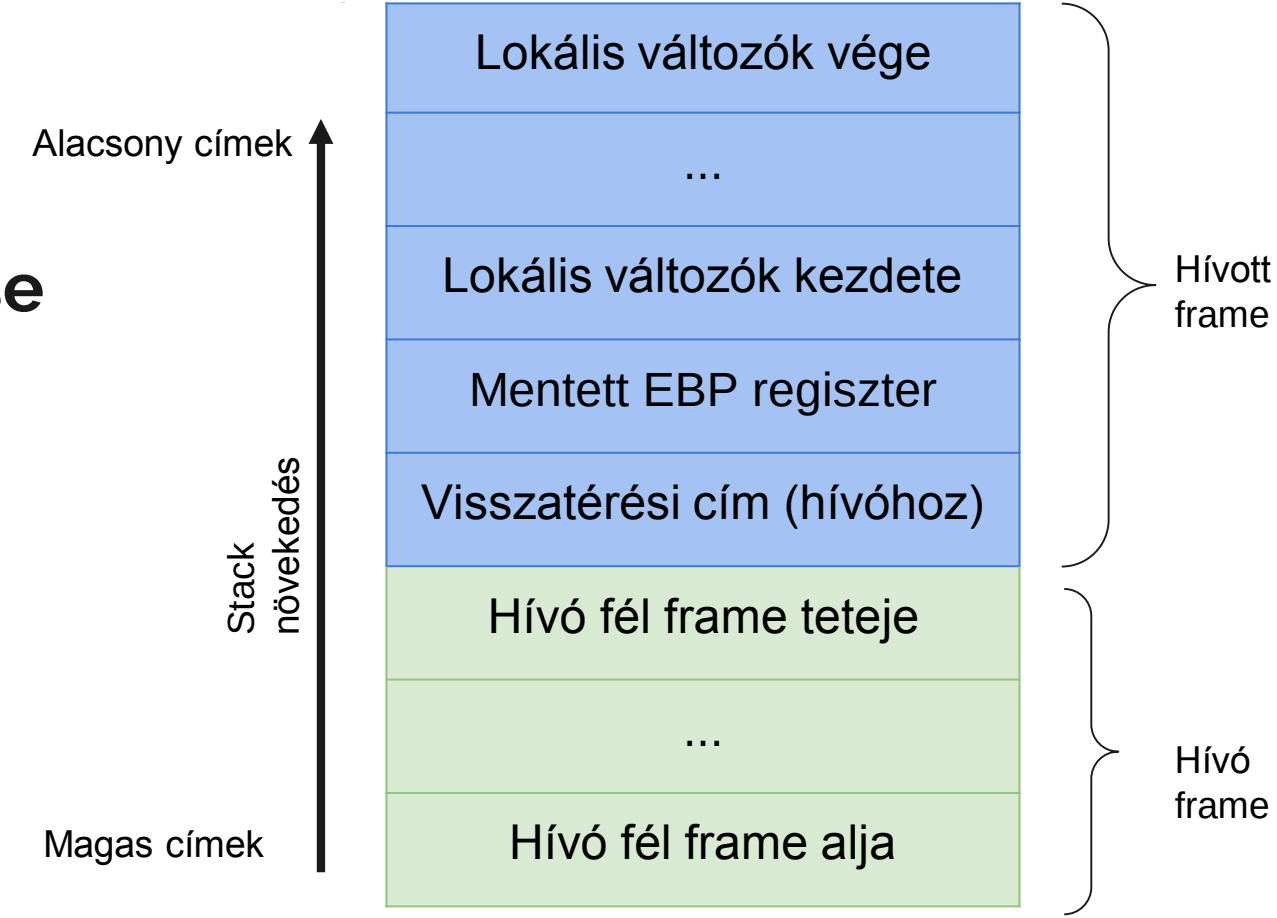
# Szubrutinhívás

# Regiszterek

- x86 32bit, AT&T
- ESP - Stack Pointer
  - A verem tetejére mutat
- EBP - Base Pointer
  - A frame kezdetére mutat
- EIP - Instruction Pointer
  - A következő végrehajtandó utasítás címét tárolja

# Verem felépítése



# Szubrutin hívás menete, hívó fél

1. A hívó elhelyezi a hívott függvény paramétereit a verembe fordított sorrendben
   a. PUSH <last argument>
   b. ...
   c. PUSH <first argument>
2. Az eljárás meghívása:
   a. CALL <function address>, amely az alábbiakat végzi:
      i. PUSH EIP
      ii. JMP <func address>

# Szubrutin hívás menete, hívott fél

3. A hívott szubrutin menti az **EBP** tartalmát
   a. **PUSH EBP**
4. Az ESP tartalmát a hívott szubrutin az EBP-be menti
   a. **MOV ESP, EBP**
5. Amennyiben vannak/kellenek lokális változók akkor azoknak a stack-en helyet kell foglalni/csinálni pl.:
   a. Ha **3** int-nek kell hely akkor az **ESP**-t **3*4** bájtal kell csökkenteni.
   b. Így az **EBP - 4** címen van az első lokális változó, az **EBP - 8** -on a második és így tovább
   c. ehhez a **SUB 3 * 4, ESP**-t lehet használni
   d. (figyelem a 3*4-et a fordító ki tudja számolni :) )

# Visszatérés szubrutin hívásból, hívott fél

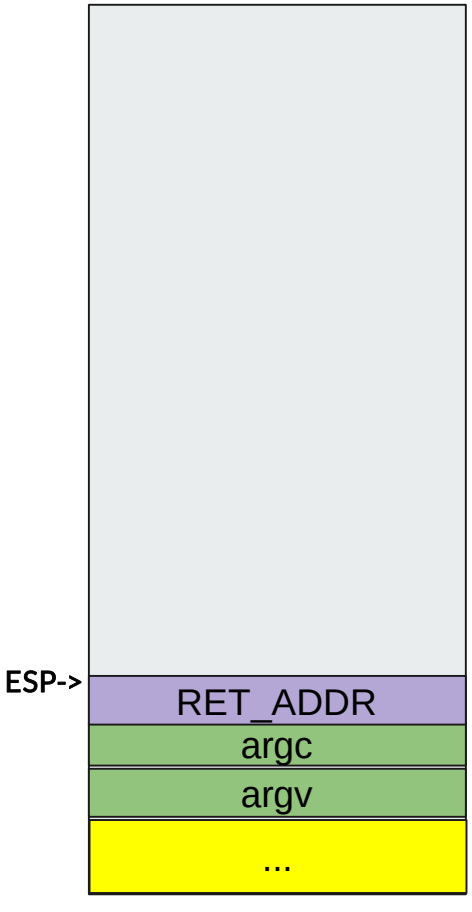
6. Az **ESP** visszaállítása
   a. A hívott szubrutin visszaállítja a függvény hívás kezdeti állapotát: **LEAVE**
      i. Amely ekvivalens a **MOV EBP, ESP ; POP EBP**
   b. 3. és 4. pont ‘visszafelé’
7. Visszatérés a hívóhoz
   a. A verembe mentett visszatérési cím az **EIP**-be kerül
      i. **POP EIP** (közvetetten a 8.b lépés végzi)
   b. **RET**

# Visszatérés szubrutin hívásból, hívó fél

8. Az argumentumokat a  hívó fél “takarítja”
    a. **ADD N*4, ESP**
        i. Ahol **N** az verembe pusholt argumentumok száma

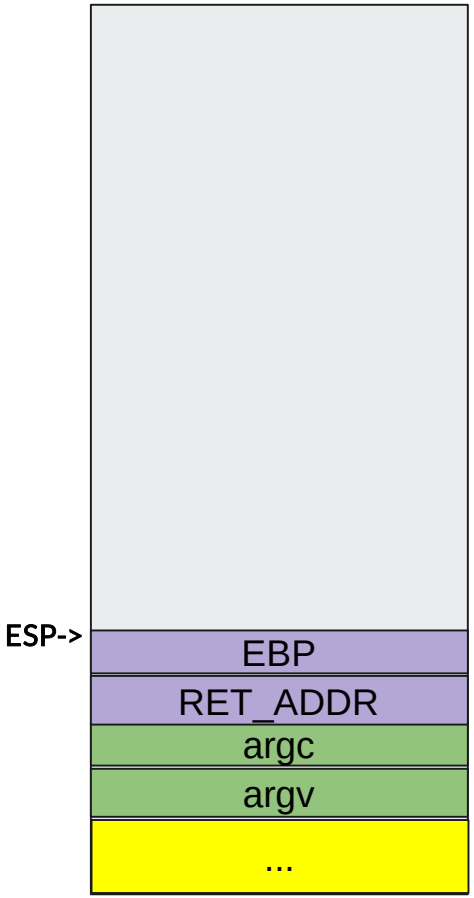
```c
int
calc (int lhs,
      int rhs)
{
  int local1 = 1 + lhs;
  int local2 = rhs - 2;

  return local1 * local2;
}

int
main (int argc,
      char **argv)
{
  int a = 5;
  int b = 6;
  int r = calc (a, b);

  return 0;
}
```

```
Dump of assembler code for function calc:
   0x000004ed <+0>:     push   %ebp
   0x000004ee <+1>:     mov    %esp,%ebp
   0x000004f0 <+3>:     sub    $0x10,%esp
   0x000004f3 <+6>:     mov    0x8(%ebp),%eax
   0x000004f6 <+9>:     add    $0x1,%eax
   0x000004f9 <+12>:    mov    %eax,-0x4(%ebp)
   0x000004fc <+15>:    mov    0xc(%ebp),%eax
   0x000004ff <+18>:    sub    $0x2,%eax
   0x00000502 <+21>:    mov    %eax,-0x8(%ebp)
   0x00000505 <+24>:    mov    -0x4(%ebp),%eax
   0x00000508 <+27>:    imul   -0x8(%ebp),%eax
   0x0000050c <+31>:    leave
   0x0000050d <+32>:    ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
   0x0000050e <+0>:     push   %ebp
   0x0000050f <+1>:     mov    %esp,%ebp
   0x00000511 <+3>:     sub    $0x10,%esp
   0x00000514 <+6>:     movl   $0x5,-0x4(%ebp)
   0x0000051b <+13>:    movl   $0x6,-0x8(%ebp)
   0x00000522 <+20>:    pushl  -0x8(%ebp)
   0x00000525 <+23>:    pushl  -0x4(%ebp)
   0x00000528 <+26>:    call   0x4ed <calc>
   0x0000052d <+31>:    add    $0x8,%esp
   0x00000530 <+34>:    mov    %eax,-0xc(%ebp)
   0x00000533 <+37>:    mov    $0x0,%eax
   0x00000538 <+42>:    leave
   0x00000539 <+43>:    ret
End of assembler dump.
```



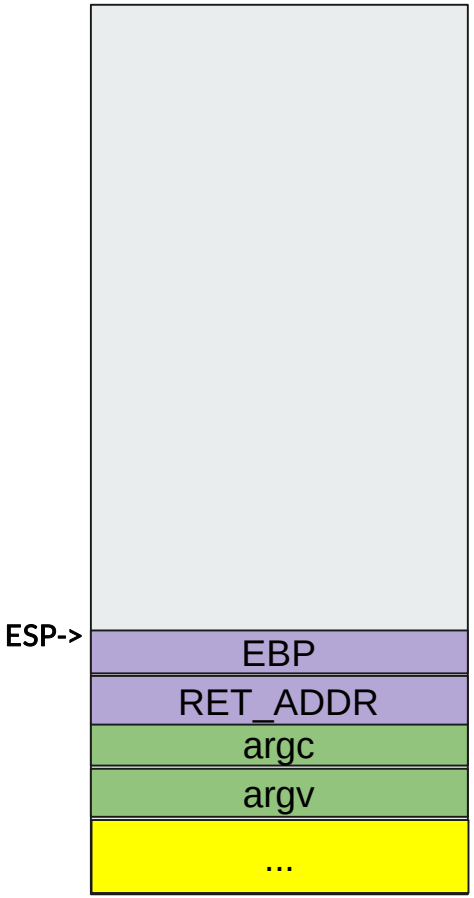

```c
int
calc (int lhs,
      int rhs)
{
  int local1 = 1 + lhs;
  int local2 = rhs - 2;

  return local1 * local2;
}

int
main (int argc,
      char **argv)
{
  int a = 5;
  int b = 6;
  int r = calc (a, b);

  return 0;
}
```

```
Dump of assembler code for function calc:
   0x000004ed <+0>:     push   %ebp
   0x000004ee <+1>:     mov    %esp,%ebp
   0x000004f0 <+3>:     sub    $0x10,%esp
   0x000004f3 <+6>:     mov    0x8(%ebp),%eax
   0x000004f6 <+9>:     add    $0x1,%eax
   0x000004f9 <+12>:    mov    %eax,-0x4(%ebp)
   0x000004fc <+15>:    mov    0xc(%ebp),%eax
   0x000004ff <+18>:    sub    $0x2,%eax
   0x00000502 <+21>:    mov    %eax,-0x8(%ebp)
   0x00000505 <+24>:    mov    -0x4(%ebp),%eax
   0x00000508 <+27>:    imul   -0x8(%ebp),%eax
   0x0000050c <+31>:    leave
   0x0000050d <+32>:    ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
   0x0000050e <+0>:     push   %ebp
   0x0000050f <+1>:     mov    %esp,%ebp
   0x00000511 <+3>:     sub    $0x10,%esp
   0x00000514 <+6>:     movl   $0x5,-0x4(%ebp)
   0x0000051b <+13>:    movl   $0x6,-0x8(%ebp)
   0x00000522 <+20>:    pushl  -0x8(%ebp)
   0x00000525 <+23>:    pushl  -0x4(%ebp)
   0x00000528 <+26>:    call   0x4ed <calc>
   0x0000052d <+31>:    add    $0x8,%esp
   0x00000530 <+34>:    mov    %eax,-0xc(%ebp)
   0x00000533 <+37>:    mov    $0x0,%eax
   0x00000538 <+42>:    leave
   0x00000539 <+43>:    ret
End of assembler dump.
```



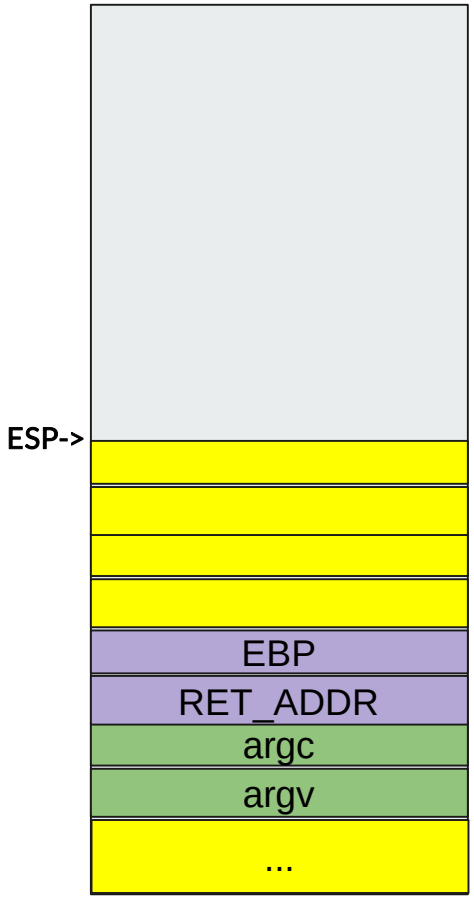

```
int
calc (int lhs,
      int rhs)
{
  int local1 = 1 + lhs;
  int local2 = rhs - 2;

  return local1 * local2;
}

int
main (int argc,
      char **argv)
{
  int a = 5;
  int b = 6;
  int r = calc (a, b);

  return 0;
}
```

```
Dump of assembler code for function calc:
   0x000004ed <+0>:     push   %ebp
   0x000004ee <+1>:     mov    %esp,%ebp
   0x000004f0 <+3>:     sub    $0x10,%esp
   0x000004f3 <+6>:     mov    0x8(%ebp),%eax
   0x000004f6 <+9>:     add    $0x1,%eax
   0x000004f9 <+12>:    mov    %eax,-0x4(%ebp)
   0x000004fc <+15>:    mov    0xc(%ebp),%eax
   0x000004ff <+18>:    sub    $0x2,%eax
   0x00000502 <+21>:    mov    %eax,-0x8(%ebp)
   0x00000505 <+24>:    mov    -0x4(%ebp),%eax
   0x00000508 <+27>:    imul   -0x8(%ebp),%eax
   0x0000050c <+31>:    leave
   0x0000050d <+32>:    ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
   0x0000050e <+0>:     push   %ebp
   0x0000050f <+1>:     mov    %esp,%ebp
   0x00000511 <+3>:     sub    $0x10,%esp
   0x00000514 <+6>:     movl   $0x5,-0x4(%ebp)
   0x0000051b <+13>:    movl   $0x6,-0x8(%ebp)
   0x00000522 <+20>:    pushl  -0x8(%ebp)
   0x00000525 <+23>:    pushl  -0x4(%ebp)
   0x00000528 <+26>:    call   0x4ed <calc>
   0x0000052d <+31>:    add    $0x8,%esp
   0x00000530 <+34>:    mov    %eax,-0xc(%ebp)
   0x00000533 <+37>:    mov    $0x0,%eax
   0x00000538 <+42>:    leave
   0x00000539 <+43>:    ret
End of assembler dump.
```



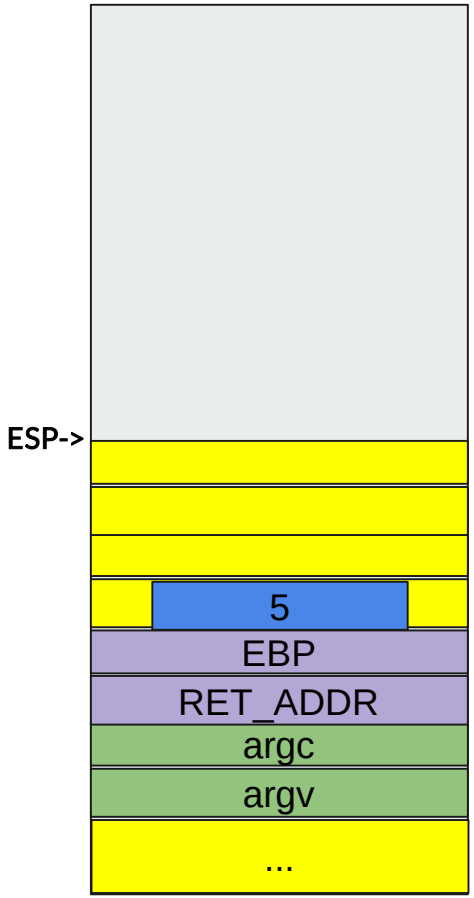

```
int
calc (int lhs,
      int rhs)
{
  int local1 = 1 + lhs;
  int local2 = rhs - 2;

  return local1 * local2;
}

int
main (int argc,
      char **argv)
{
  int a = 5;
  int b = 6;
  int r = calc (a, b);

  return 0;
}
```

```
Dump of assembler code for function calc:
   0x000004ed <+0>:     push   %ebp
   0x000004ee <+1>:     mov    %esp,%ebp
   0x000004f0 <+3>:     sub    $0x10,%esp
   0x000004f3 <+6>:     mov    0x8(%ebp),%eax
   0x000004f6 <+9>:     add    $0x1,%eax
   0x000004f9 <+12>:    mov    %eax,-0x4(%ebp)
   0x000004fc <+15>:    mov    0xc(%ebp),%eax
   0x000004ff <+18>:    sub    $0x2,%eax
   0x00000502 <+21>:    mov    %eax,-0x8(%ebp)
   0x00000505 <+24>:    mov    -0x4(%ebp),%eax
   0x00000508 <+27>:    imul   -0x8(%ebp),%eax
   0x0000050c <+31>:    leave
   0x0000050d <+32>:    ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
   0x0000050e <+0>:     push   %ebp
   0x0000050f <+1>:     mov    %esp,%ebp
   0x00000511 <+3>:     sub    $0x10,%esp
   0x00000514 <+6>:     movl   $0x5,-0x4(%ebp)
   0x0000051b <+13>:    movl   $0x6,-0x8(%ebp)
   0x00000522 <+20>:    pushl  -0x8(%ebp)
   0x00000525 <+23>:    pushl  -0x4(%ebp)
   0x00000528 <+26>:    call   0x4ed <calc>
   0x0000052d <+31>:    add    $0x8,%esp
   0x00000530 <+34>:    mov    %eax,-0xc(%ebp)
   0x00000533 <+37>:    mov    $0x0,%eax
   0x00000538 <+42>:    leave
   0x00000539 <+43>:    ret
End of assembler dump.
```



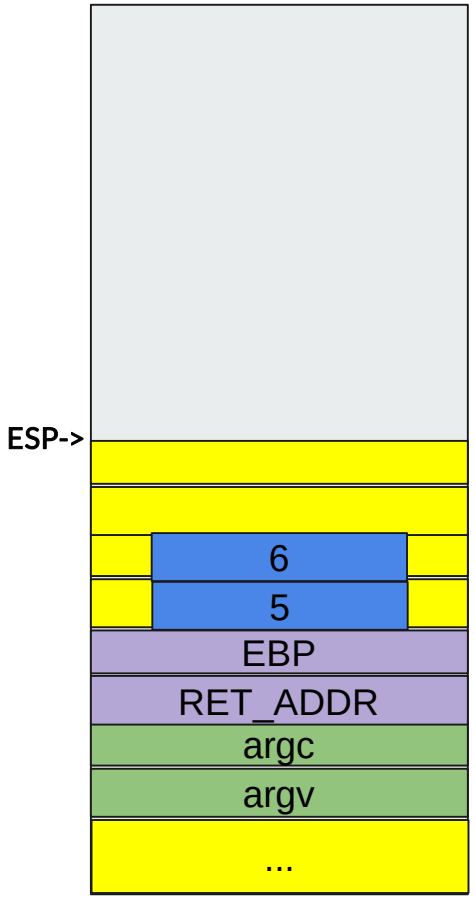

```
int
calc (int lhs,
      int rhs)
{
  int local1 = 1 + lhs;
  int local2 = rhs - 2;

  return local1 * local2;
}

int
main (int argc,
      char **argv)
{
  int a = 5;
  int b = 6;
  int r = calc (a, b);

  return 0;
}
```

```
Dump of assembler code for function calc:
   0x000004ed <+0>:     push   %ebp
   0x000004ee <+1>:     mov    %esp,%ebp
   0x000004f0 <+3>:     sub    $0x10,%esp
   0x000004f3 <+6>:     mov    0x8(%ebp),%eax
   0x000004f6 <+9>:     add    $0x1,%eax
   0x000004f9 <+12>:    mov    %eax,-0x4(%ebp)
   0x000004fc <+15>:    mov    0xc(%ebp),%eax
   0x000004ff <+18>:    sub    $0x2,%eax
   0x00000502 <+21>:    mov    %eax,-0x8(%ebp)
   0x00000505 <+24>:    mov    -0x4(%ebp),%eax
   0x00000508 <+27>:    imul   -0x8(%ebp),%eax
   0x0000050c <+31>:    leave
   0x0000050d <+32>:    ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
   0x0000050e <+0>:     push   %ebp
   0x0000050f <+1>:     mov    %esp,%ebp
   0x00000511 <+3>:     sub    $0x10,%esp
   0x00000514 <+6>:     movl   $0x5,-0x4(%ebp)
   0x0000051b <+13>:    movl   $0x6,-0x8(%ebp)
   0x00000522 <+20>:    pushl  -0x8(%ebp)
   0x00000525 <+23>:    pushl  -0x4(%ebp)
   0x00000528 <+26>:    call   0x4ed <calc>
   0x0000052d <+31>:    add    $0x8,%esp
   0x00000530 <+34>:    mov    %eax,-0xc(%ebp)
   0x00000533 <+37>:    mov    $0x0,%eax
   0x00000538 <+42>:    leave
   0x00000539 <+43>:    ret
End of assembler dump.
```



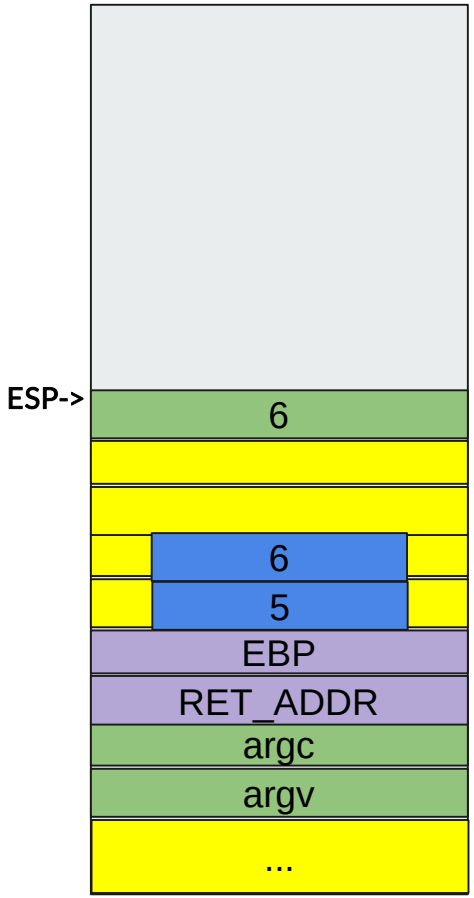

```
int
calc (int lhs,
      int rhs)
{
  int local1 = 1 + lhs;
  int local2 = rhs - 2;

  return local1 * local2;
}

int
main (int argc,
      char **argv)
{
  int a = 5;
  int b = 6;
  int r = calc (a, b);

  return 0;
}
```

```
Dump of assembler code for function calc:
   0x000004ed <+0>:     push   %ebp
   0x000004ee <+1>:     mov    %esp,%ebp
   0x000004f0 <+3>:     sub    $0x10,%esp
   0x000004f3 <+6>:     mov    0x8(%ebp),%eax
   0x000004f6 <+9>:     add    $0x1,%eax
   0x000004f9 <+12>:    mov    %eax,-0x4(%ebp)
   0x000004fc <+15>:    mov    0xc(%ebp),%eax
   0x000004ff <+18>:    sub    $0x2,%eax
   0x00000502 <+21>:    mov    %eax,-0x8(%ebp)
   0x00000505 <+24>:    mov    -0x4(%ebp),%eax
   0x00000508 <+27>:    imul   -0x8(%ebp),%eax
   0x0000050c <+31>:    leave
   0x0000050d <+32>:    ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
   0x0000050e <+0>:     push   %ebp
   0x0000050f <+1>:     mov    %esp,%ebp
   0x00000511 <+3>:     sub    $0x10,%esp
   0x00000514 <+6>:     movl   $0x5,-0x4(%ebp)
   0x0000051b <+13>:    movl   $0x6,-0x8(%ebp)
   0x00000522 <+20>:    pushl  -0x8(%ebp)
   0x00000525 <+23>:    pushl  -0x4(%ebp)
   0x00000528 <+26>:    call   0x4ed <calc>
   0x0000052d <+31>:    add    $0x8,%esp
   0x00000530 <+34>:    mov    %eax,-0xc(%ebp)
   0x00000533 <+37>:    mov    $0x0,%eax
   0x00000538 <+42>:    leave
   0x00000539 <+43>:    ret
End of assembler dump.
```



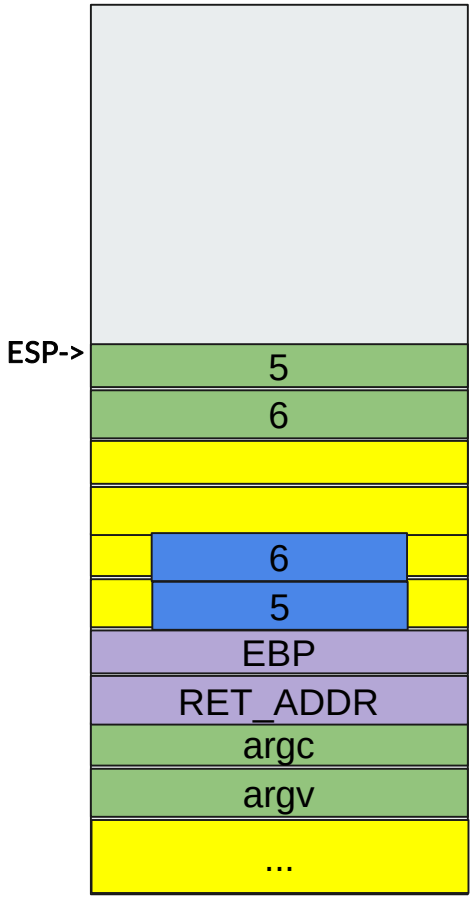

```
int
calc (int lhs,
      int rhs)
{
  int local1 = 1 + lhs;
  int local2 = rhs - 2;

  return local1 * local2;
}

int
main (int argc,
      char **argv)
{
  int a = 5;
  int b = 6;
  int r = calc (a, b);

  return 0;
}
```

```
Dump of assembler code for function calc:
   0x000004ed <+0>:     push   %ebp
   0x000004ee <+1>:     mov    %esp,%ebp
   0x000004f0 <+3>:     sub    $0x10,%esp
   0x000004f3 <+6>:     mov    0x8(%ebp),%eax
   0x000004f6 <+9>:     add    $0x1,%eax
   0x000004f9 <+12>:    mov    %eax,-0x4(%ebp)
   0x000004fc <+15>:    mov    0xc(%ebp),%eax
   0x000004ff <+18>:    sub    $0x2,%eax
   0x00000502 <+21>:    mov    %eax,-0x8(%ebp)
   0x00000505 <+24>:    mov    -0x4(%ebp),%eax
   0x00000508 <+27>:    imul   -0x8(%ebp),%eax
   0x0000050c <+31>:    leave
   0x0000050d <+32>:    ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
   0x0000050e <+0>:     push   %ebp
   0x0000050f <+1>:     mov    %esp,%ebp
   0x00000511 <+3>:     sub    $0x10,%esp
   0x00000514 <+6>:     movl   $0x5,-0x4(%ebp)
   0x0000051b <+13>:    movl   $0x6,-0x8(%ebp)
   0x00000522 <+20>:    pushl  -0x8(%ebp)
   0x00000525 <+23>:    pushl  -0x4(%ebp)
   0x00000528 <+26>:    call   0x4ed <calc>
   0x0000052d <+31>:    add    $0x8,%esp
   0x00000530 <+34>:    mov    %eax,-0xc(%ebp)
   0x00000533 <+37>:    mov    $0x0,%eax
   0x00000538 <+42>:    leave
   0x00000539 <+43>:    ret
End of assembler dump.
```



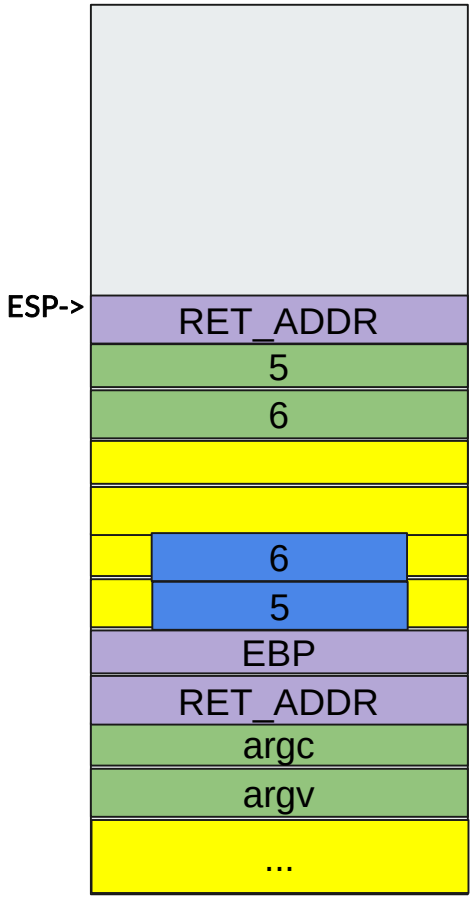

```
int
calc (int lhs,
      int rhs)
{
  int local1 = 1 + lhs;
  int local2 = rhs - 2;

  return local1 * local2;
}

int
main (int argc,
      char **argv)
{
  int a = 5;
  int b = 6;
  int r = calc (a, b);

  return 0;
}
```

```
Dump of assembler code for function calc:
   0x000004ed <+0>:     push   %ebp
   0x000004ee <+1>:     mov    %esp,%ebp
   0x000004f0 <+3>:     sub    $0x10,%esp
   0x000004f3 <+6>:     mov    0x8(%ebp),%eax
   0x000004f6 <+9>:     add    $0x1,%eax
   0x000004f9 <+12>:    mov    %eax,-0x4(%ebp)
   0x000004fc <+15>:    mov    0xc(%ebp),%eax
   0x000004ff <+18>:    sub    $0x2,%eax
   0x00000502 <+21>:    mov    %eax,-0x8(%ebp)
   0x00000505 <+24>:    mov    -0x4(%ebp),%eax
   0x00000508 <+27>:    imul   -0x8(%ebp),%eax
   0x0000050c <+31>:    leave
   0x0000050d <+32>:    ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
   0x0000050e <+0>:     push   %ebp
   0x0000050f <+1>:     mov    %esp,%ebp
   0x00000511 <+3>:     sub    $0x10,%esp
   0x00000514 <+6>:     movl   $0x5,-0x4(%ebp)
   0x0000051b <+13>:    movl   $0x6,-0x8(%ebp)
   0x00000522 <+20>:    pushl  -0x8(%ebp)
   0x00000525 <+23>:    pushl  -0x4(%ebp)
   0x00000528 <+26>:    call   0x4ed <calc>
   0x0000052d <+31>:    add    $0x8,%esp
   0x00000530 <+34>:    mov    %eax,-0xc(%ebp)
   0x00000533 <+37>:    mov    $0x0,%eax
   0x00000538 <+42>:    leave
   0x00000539 <+43>:    ret
End of assembler dump.
```



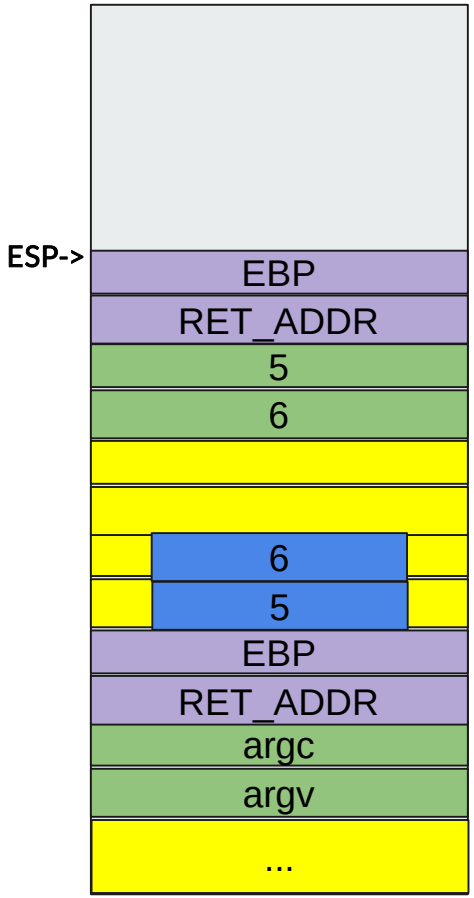

```
int
calc (int lhs,
      int rhs)
{
  int local1 = 1 + lhs;
  int local2 = rhs - 2;

  return local1 * local2;
}

int
main (int argc,
      char **argv)
{
  int a = 5;
  int b = 6;
  int r = calc (a, b);

  return 0;
}
```

```
Dump of assembler code for function calc:
   0x000004ed <+0>:     push   %ebp
   0x000004ee <+1>:     mov    %esp,%ebp
   0x000004f0 <+3>:     sub    $0x10,%esp
   0x000004f3 <+6>:     mov    0x8(%ebp),%eax
   0x000004f6 <+9>:     add    $0x1,%eax
   0x000004f9 <+12>:    mov    %eax,-0x4(%ebp)
   0x000004fc <+15>:    mov    0xc(%ebp),%eax
   0x000004ff <+18>:    sub    $0x2,%eax
   0x00000502 <+21>:    mov    %eax,-0x8(%ebp)
   0x00000505 <+24>:    mov    -0x4(%ebp),%eax
   0x00000508 <+27>:    imul   -0x8(%ebp),%eax
   0x0000050c <+31>:    leave
   0x0000050d <+32>:    ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
   0x0000050e <+0>:     push   %ebp
   0x0000050f <+1>:     mov    %esp,%ebp
   0x00000511 <+3>:     sub    $0x10,%esp
   0x00000514 <+6>:     movl   $0x5,-0x4(%ebp)
   0x0000051b <+13>:    movl   $0x6,-0x8(%ebp)
   0x00000522 <+20>:    pushl  -0x8(%ebp)
   0x00000525 <+23>:    pushl  -0x4(%ebp)
   0x00000528 <+26>:    call   0x4ed <calc>
   0x0000052d <+31>:    add    $0x8,%esp
   0x00000530 <+34>:    mov    %eax,-0xc(%ebp)
   0x00000533 <+37>:    mov    $0x0,%eax
   0x00000538 <+42>:    leave
   0x00000539 <+43>:    ret
End of assembler dump.
```



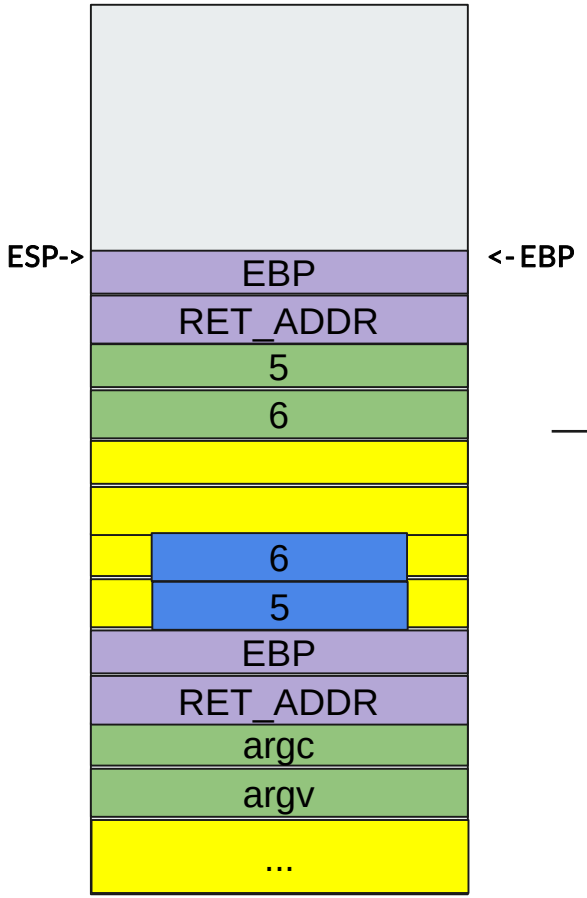

```
int
calc (int lhs,
      int rhs)
{
  int local1 = 1 + lhs;
  int local2 = rhs - 2;

  return local1 * local2;
}

int
main (int argc,
      char **argv)
{
  int a = 5;
  int b = 6;
  int r = calc (a, b);

  return 0;
}
```

```
Dump of assembler code for function calc:
   0x000004ed <+0>:     push   %ebp
   0x000004ee <+1>:     mov    %esp,%ebp
   0x000004f0 <+3>:     sub    $0x10,%esp
   0x000004f3 <+6>:     mov    0x8(%ebp),%eax
   0x000004f6 <+9>:     add    $0x1,%eax
   0x000004f9 <+12>:    mov    %eax,-0x4(%ebp)
   0x000004fc <+15>:    mov    0xc(%ebp),%eax
   0x000004ff <+18>:    sub    $0x2,%eax
   0x00000502 <+21>:    mov    %eax,-0x8(%ebp)
   0x00000505 <+24>:    mov    -0x4(%ebp),%eax
   0x00000508 <+27>:    imul   -0x8(%ebp),%eax
   0x0000050c <+31>:    leave
   0x0000050d <+32>:    ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
   0x0000050e <+0>:     push   %ebp
   0x0000050f <+1>:     mov    %esp,%ebp
   0x00000511 <+3>:     sub    $0x10,%esp
   0x00000514 <+6>:     movl   $0x5,-0x4(%ebp)
   0x0000051b <+13>:    movl   $0x6,-0x8(%ebp)
   0x00000522 <+20>:    pushl  -0x8(%ebp)
   0x00000525 <+23>:    pushl  -0x4(%ebp)
   0x00000528 <+26>:    call   0x4ed <calc>
   0x0000052d <+31>:    add    $0x8,%esp
   0x00000530 <+34>:    mov    %eax,-0xc(%ebp)
   0x00000533 <+37>:    mov    $0x0,%eax
   0x00000538 <+42>:    leave
   0x00000539 <+43>:    ret
End of assembler dump.
```



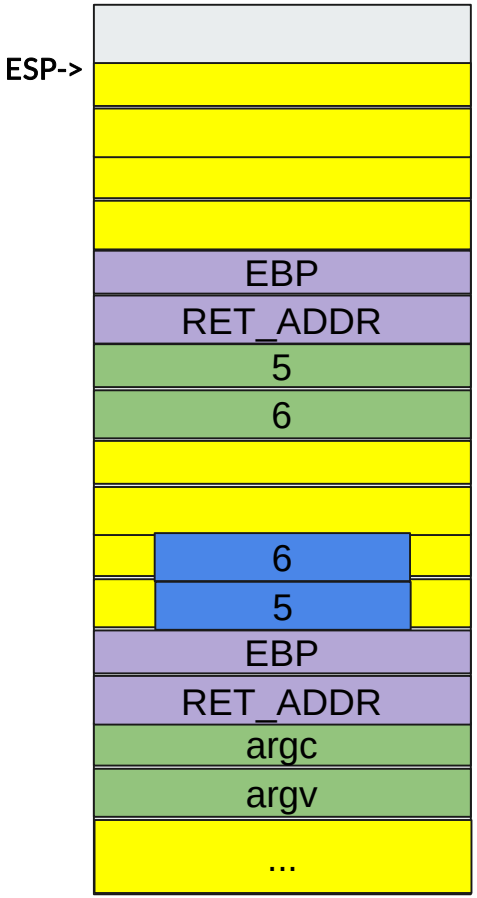

```
int
calc (int lhs,
      int rhs)
{
  int local1 = 1 + lhs;
  int local2 = rhs - 2;

  return local1 * local2;
}

int
main (int argc,
      char **argv)
{
  int a = 5;
  int b = 6;
  int r = calc (a, b);

  return 0;
}
```

```
Dump of assembler code for function calc:
   0x000004ed <+0>:     push   %ebp
   0x000004ee <+1>:     mov    %esp,%ebp
   0x000004f0 <+3>:     sub    $0x10,%esp
   0x000004f3 <+6>:     mov    0x8(%ebp),%eax
   0x000004f6 <+9>:     add    $0x1,%eax
   0x000004f9 <+12>:    mov    %eax,-0x4(%ebp)
   0x000004fc <+15>:    mov    0xc(%ebp),%eax
   0x000004ff <+18>:    sub    $0x2,%eax
   0x00000502 <+21>:    mov    %eax,-0x8(%ebp)
   0x00000505 <+24>:    mov    -0x4(%ebp),%eax
   0x00000508 <+27>:    imul   -0x8(%ebp),%eax
   0x0000050c <+31>:    leave
   0x0000050d <+32>:    ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
   0x0000050e <+0>:     push   %ebp
   0x0000050f <+1>:     mov    %esp,%ebp
   0x00000511 <+3>:     sub    $0x10,%esp
   0x00000514 <+6>:     movl   $0x5,-0x4(%ebp)
   0x0000051b <+13>:    movl   $0x6,-0x8(%ebp)
   0x00000522 <+20>:    pushl  -0x8(%ebp)
   0x00000525 <+23>:    pushl  -0x4(%ebp)
   0x00000528 <+26>:    call   0x4ed <calc>
   0x0000052d <+31>:    add    $0x8,%esp
   0x00000530 <+34>:    mov    %eax,-0xc(%ebp)
   0x00000533 <+37>:    mov    $0x0,%eax
   0x00000538 <+42>:    leave
   0x00000539 <+43>:    ret
End of assembler dump.
```



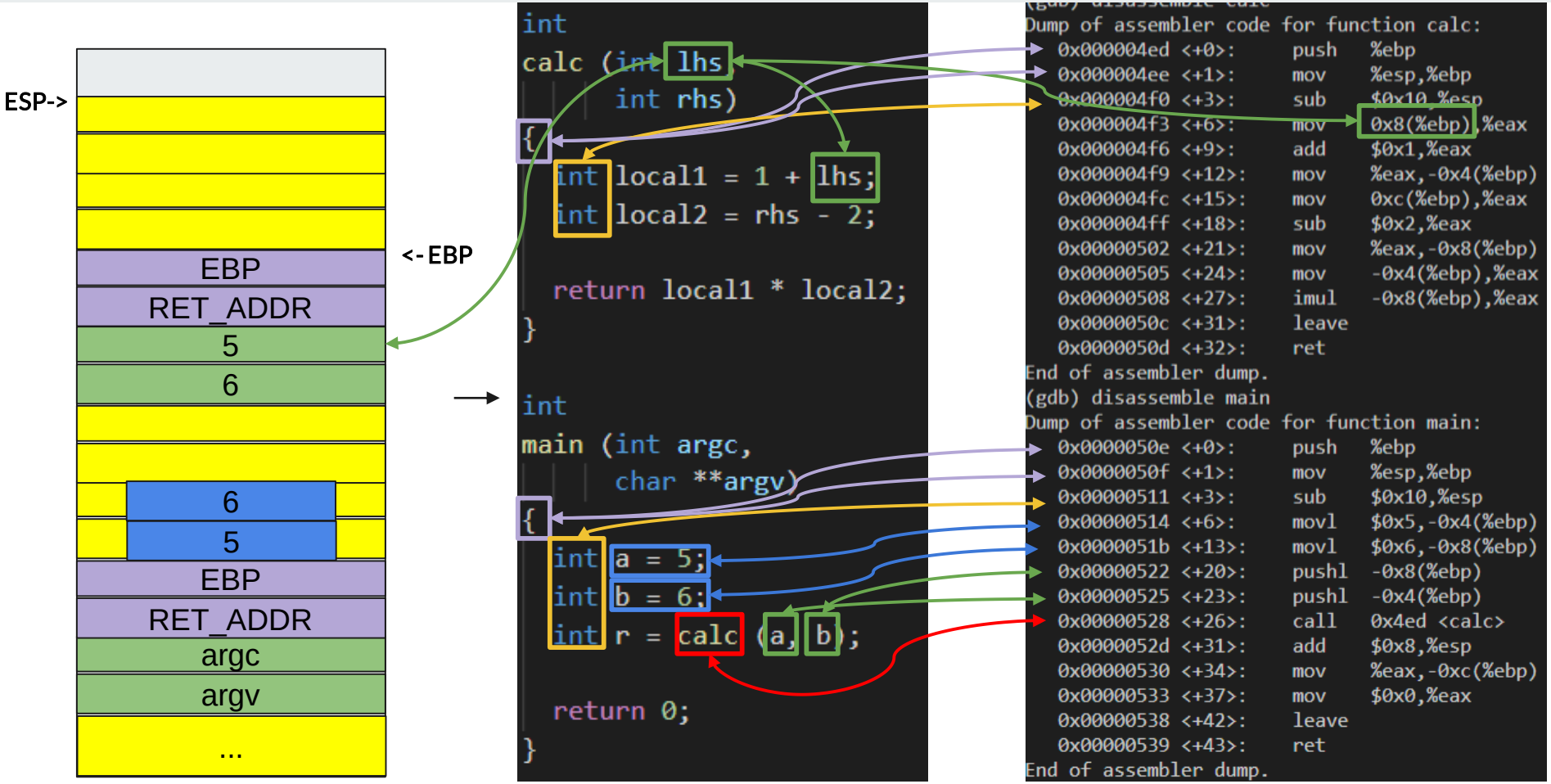

```
int
calc (int lhs,
      int rhs)
{
  int local1 = 1 + lhs;
  int local2 = rhs - 2;

  return local1 * local2;
}

int
main (int argc,
      char **argv)
{
  int a = 5;
  int b = 6;
  int r = calc (a, b);

  return 0;
}
```

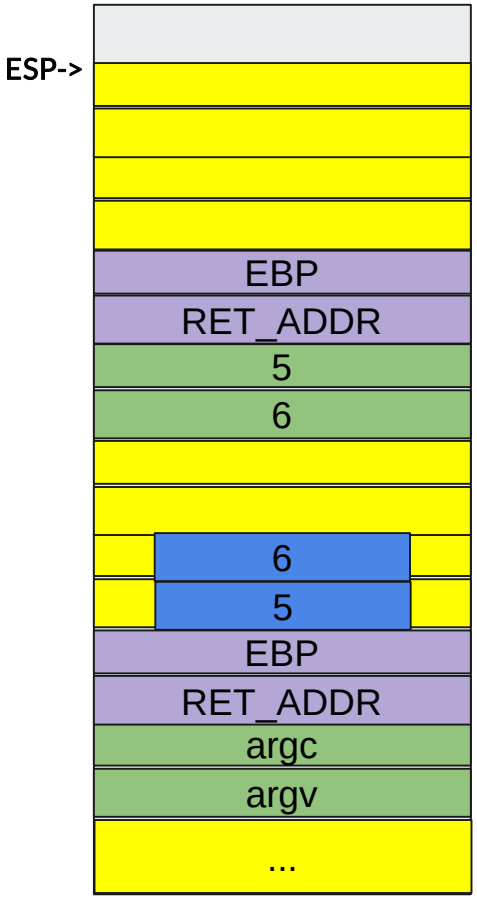
```
Dump of assembler code for function calc:
   0x000004ed <+0>:     push   %ebp
   0x000004ee <+1>:     mov    %esp,%ebp
   0x000004f0 <+3>:     sub    $0x10,%esp
   0x000004f3 <+6>:     mov    0x8(%ebp),%eax
   0x000004f6 <+9>:     add    $0x1,%eax
   0x000004f9 <+12>:    mov    %eax,-0x4(%ebp)
   0x000004fc <+15>:    mov    0xc(%ebp),%eax
   0x000004ff <+18>:    sub    $0x2,%eax
   0x00000502 <+21>:    mov    %eax,-0x8(%ebp)
   0x00000505 <+24>:    mov    -0x4(%ebp),%eax
   0x00000508 <+27>:    imul   -0x8(%ebp),%eax
   0x0000050c <+31>:    leave
   0x0000050d <+32>:    ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
   0x0000050e <+0>:     push   %ebp
   0x0000050f <+1>:     mov    %esp,%ebp
   0x00000511 <+3>:     sub    $0x10,%esp
   0x00000514 <+6>:     movl   $0x5,-0x4(%ebp)
   0x0000051b <+13>:    movl   $0x6,-0x8(%ebp)
   0x00000522 <+20>:    pushl  -0x8(%ebp)
   0x00000525 <+23>:    pushl  -0x4(%ebp)
   0x00000528 <+26>:    call   0x4ed <calc>
   0x0000052d <+31>:    add    $0x8,%esp
   0x00000530 <+34>:    mov    %eax,-0xc(%ebp)
   0x00000533 <+37>:    mov    $0x0,%eax
   0x00000538 <+42>:    leave
   0x00000539 <+43>:    ret
End of assembler dump.
```




```
int
calc (int lhs,
      int rhs)
{
  int local1 = 1 + lhs;
  int local2 = rhs - 2;

  return local1 * local2;
}

int
main (int argc,
      char **argv)
{
  int a = 5;
  int b = 6;
  int r = calc (a, b);

  return 0;
}
```

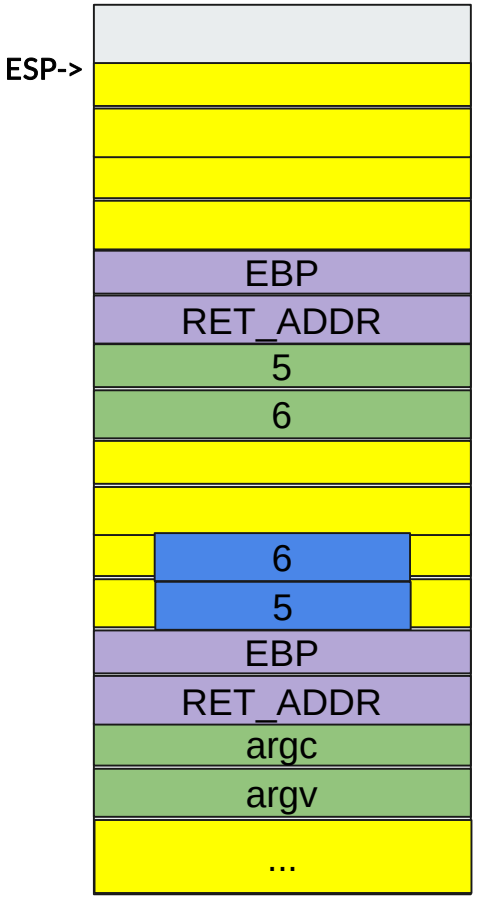
```
Dump of assembler code for function calc:
   0x000004ed <+0>:     push   %ebp
   0x000004ee <+1>:     mov    %esp,%ebp
   0x000004f0 <+3>:     sub    $0x10,%esp
   0x000004f3 <+6>:     mov    0x8(%ebp),%eax
   0x000004f6 <+9>:     add    $0x1,%eax
   0x000004f9 <+12>:    mov    %eax,-0x4(%ebp)
   0x000004fc <+15>:    mov    0xc(%ebp),%eax
   0x000004ff <+18>:    sub    $0x2,%eax
   0x00000502 <+21>:    mov    %eax,-0x8(%ebp)
   0x00000505 <+24>:    mov    -0x4(%ebp),%eax
   0x00000508 <+27>:    imul   -0x8(%ebp),%eax
   0x0000050c <+31>:    leave
   0x0000050d <+32>:    ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
   0x0000050e <+0>:     push   %ebp
   0x0000050f <+1>:     mov    %esp,%ebp
   0x00000511 <+3>:     sub    $0x10,%esp
   0x00000514 <+6>:     movl   $0x5,-0x4(%ebp)
   0x0000051b <+13>:    movl   $0x6,-0x8(%ebp)
   0x00000522 <+20>:    pushl  -0x8(%ebp)
   0x00000525 <+23>:    pushl  -0x4(%ebp)
   0x00000528 <+26>:    call   0x4ed <calc>
   0x0000052d <+31>:    add    $0x8,%esp
   0x00000530 <+34>:    mov    %eax,-0xc(%ebp)
   0x00000533 <+37>:    mov    $0x0,%eax
   0x00000538 <+42>:    leave
   0x00000539 <+43>:    ret
End of assembler dump.
```


ESP->
<- EBP
EBP
RET_ADDR
5
6
6
5
EBP
RET_ADDR
argc
argv
...
int
calc (int lhs,
int rhs)
{
int local1 = 1 + lhs;
int local2 = rhs - 2;
return local1 * local2;
}
int
main (int argc,
char **argv)
{
int a = 5;
int b = 6;
int r = calc (a, b);
return 0;
}
Dump of assembler code for function calc:
0x000004ed <+0>: push %ebp
0x000004ee <+1>: mov %esp,%ebp
0x000004f0 <+3>: sub $0x10,%esp
0x000004f3 <+6>: mov 0x8(%ebp),%eax
0x000004f6 <+9>: add $0x1,%eax
0x000004f9 <+12>: mov %eax,-0x4(%ebp)
0x000004fc <+15>: mov 0xc(%ebp),%eax
0x000004ff <+18>: sub $0x2,%eax
0x00000502 <+21>: mov %eax,-0x8(%ebp)
0x00000505 <+24>: mov -0x4(%ebp),%eax
0x00000508 <+27>: imul -0x8(%ebp),%eax
0x0000050c <+31>: leave
0x0000050d <+32>: ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
0x0000050e <+0>: push %ebp
0x0000050f <+1>: mov %esp,%ebp
0x00000511 <+3>: sub $0x10,%esp
0x00000514 <+6>: movl $0x5,-0x4(%ebp)
0x0000051b <+13>: movl $0x6,-0x8(%ebp)
0x00000522 <+20>: pushl -0x8(%ebp)
0x00000525 <+23>: pushl -0x4(%ebp)
0x00000528 <+26>: call 0x4ed <calc>
0x0000052d <+31>: add $0x8,%esp
0x00000530 <+34>: mov %eax,-0xc(%ebp)
0x00000533 <+37>: mov $0x0,%eax
0x00000538 <+42>: leave
0x00000539 <+43>: ret
End of assembler dump.



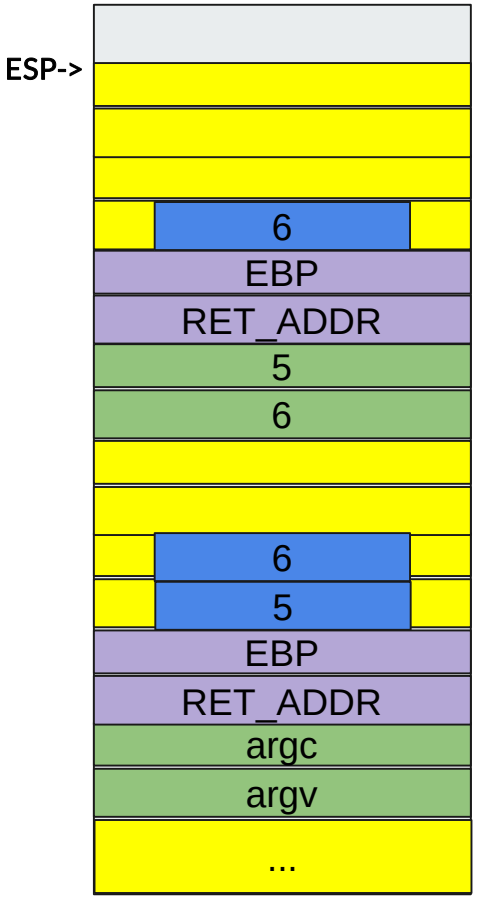

```
int
calc (int lhs,
      int rhs)
{
  int local1 = 1 + lhs;
  int local2 = rhs - 2;

  return local1 * local2;
}

int
main (int argc,
      char **argv)
{
  int a = 5;
  int b = 6;
  int r = calc (a, b);

  return 0;
}
```

```
Dump of assembler code for function calc:
   0x000004ed <+0>:     push   %ebp
   0x000004ee <+1>:     mov    %esp,%ebp
   0x000004f0 <+3>:     sub    $0x10,%esp
   0x000004f3 <+6>:     mov    0x8(%ebp),%eax
   0x000004f6 <+9>:     add    $0x1,%eax
   0x000004f9 <+12>:    mov    %eax,-0x4(%ebp)
   0x000004fc <+15>:    mov    0xc(%ebp),%eax
   0x000004ff <+18>:    sub    $0x2,%eax
   0x00000502 <+21>:    mov    %eax,-0x8(%ebp)
   0x00000505 <+24>:    mov    -0x4(%ebp),%eax
   0x00000508 <+27>:    imul   -0x8(%ebp),%eax
   0x0000050c <+31>:    leave
   0x0000050d <+32>:    ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
   0x0000050e <+0>:     push   %ebp
   0x0000050f <+1>:     mov    %esp,%ebp
   0x00000511 <+3>:     sub    $0x10,%esp
   0x00000514 <+6>:     movl   $0x5,-0x4(%ebp)
   0x0000051b <+13>:    movl   $0x6,-0x8(%ebp)
   0x00000522 <+20>:    pushl  -0x8(%ebp)
   0x00000525 <+23>:    pushl  -0x4(%ebp)
   0x00000528 <+26>:    call   0x4ed <calc>
   0x0000052d <+31>:    add    $0x8,%esp
   0x00000530 <+34>:    mov    %eax,-0xc(%ebp)
   0x00000533 <+37>:    mov    $0x0,%eax
   0x00000538 <+42>:    leave
   0x00000539 <+43>:    ret
End of assembler dump.
```



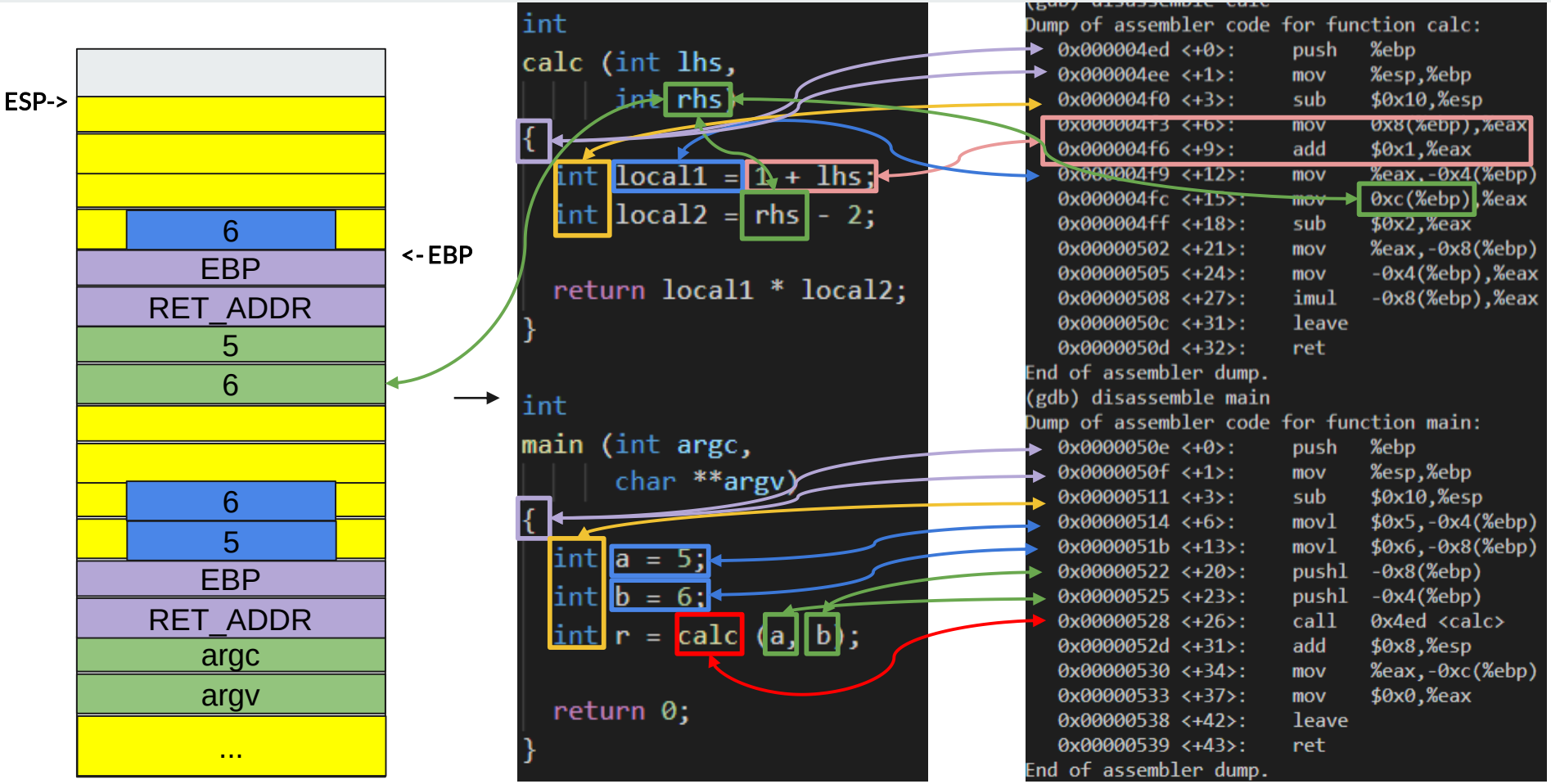

```
int
calc (int lhs,
      int rhs)
{
  int local1 = 1 + lhs;
  int local2 = rhs - 2;

  return local1 * local2;
}

int
main (int argc,
      char **argv)
{
  int a = 5;
  int b = 6;
  int r = calc (a, b);

  return 0;
}
```

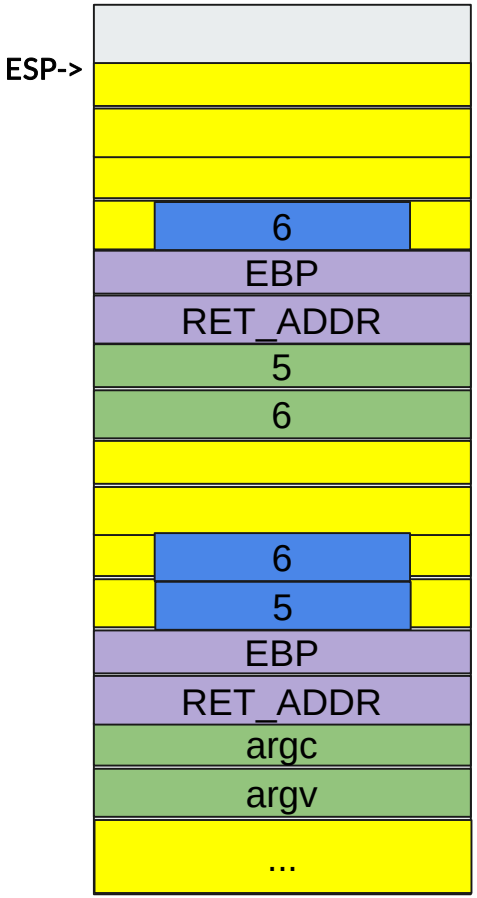
```
Dump of assembler code for function calc:
   0x000004ed <+0>:     push   %ebp
   0x000004ee <+1>:     mov    %esp,%ebp
   0x000004f0 <+3>:     sub    $0x10,%esp
   0x000004f3 <+6>:     mov    0x8(%ebp),%eax
   0x000004f6 <+9>:     add    $0x1,%eax
   0x000004f9 <+12>:    mov    %eax,-0x4(%ebp)
   0x000004fc <+15>:    mov    0xc(%ebp),%eax
   0x000004ff <+18>:    sub    $0x2,%eax
   0x00000502 <+21>:    mov    %eax,-0x8(%ebp)
   0x00000505 <+24>:    mov    -0x4(%ebp),%eax
   0x00000508 <+27>:    imul   -0x8(%ebp),%eax
   0x0000050c <+31>:    leave
   0x0000050d <+32>:    ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
   0x0000050e <+0>:     push   %ebp
   0x0000050f <+1>:     mov    %esp,%ebp
   0x00000511 <+3>:     sub    $0x10,%esp
   0x00000514 <+6>:     movl   $0x5,-0x4(%ebp)
   0x0000051b <+13>:    movl   $0x6,-0x8(%ebp)
   0x00000522 <+20>:    pushl  -0x8(%ebp)
   0x00000525 <+23>:    pushl  -0x4(%ebp)
   0x00000528 <+26>:    call   0x4ed <calc>
   0x0000052d <+31>:    add    $0x8,%esp
   0x00000530 <+34>:    mov    %eax,-0xc(%ebp)
   0x00000533 <+37>:    mov    $0x0,%eax
   0x00000538 <+42>:    leave
   0x00000539 <+43>:    ret
End of assembler dump.
```




```
int
calc (int lhs,
      int rhs)
{
  int local1 = 1 + lhs;
  int local2 = rhs - 2;

  return local1 * local2;
}

int
main (int argc,
      char **argv)
{
  int a = 5;
  int b = 6;
  int r = calc (a, b);

  return 0;
}
```

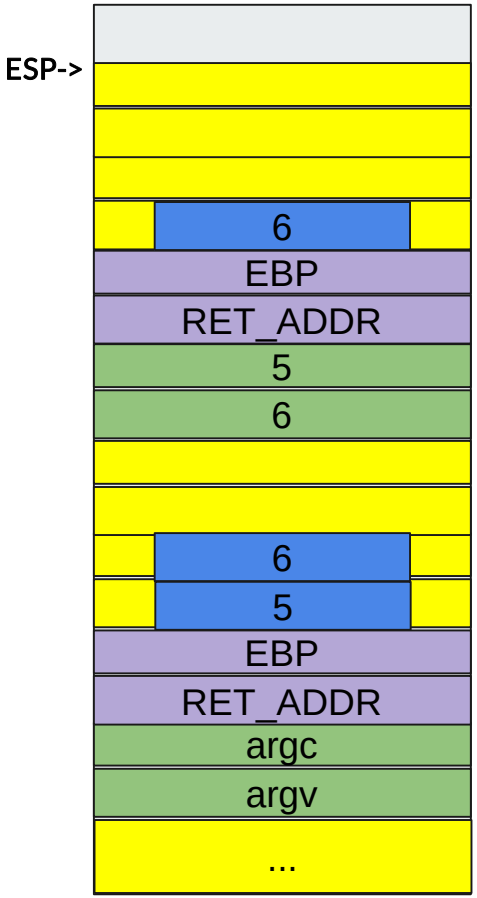
```
Dump of assembler code for function calc:
   0x000004ed <+0>:     push   %ebp
   0x000004ee <+1>:     mov    %esp,%ebp
   0x000004f0 <+3>:     sub    $0x10,%esp
   0x000004f3 <+6>:     mov    0x8(%ebp),%eax
   0x000004f6 <+9>:     add    $0x1,%eax
   0x000004f9 <+12>:    mov    %eax,-0x4(%ebp)
   0x000004fc <+15>:    mov    0xc(%ebp),%eax
   0x000004ff <+18>:    sub    $0x2,%eax
   0x00000502 <+21>:    mov    %eax,-0x8(%ebp)
   0x00000505 <+24>:    mov    -0x4(%ebp),%eax
   0x00000508 <+27>:    imul   -0x8(%ebp),%eax
   0x0000050c <+31>:    leave
   0x0000050d <+32>:    ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
   0x0000050e <+0>:     push   %ebp
   0x0000050f <+1>:     mov    %esp,%ebp
   0x00000511 <+3>:     sub    $0x10,%esp
   0x00000514 <+6>:     movl   $0x5,-0x4(%ebp)
   0x0000051b <+13>:    movl   $0x6,-0x8(%ebp)
   0x00000522 <+20>:    pushl  -0x8(%ebp)
   0x00000525 <+23>:    pushl  -0x4(%ebp)
   0x00000528 <+26>:    call   0x4ed <calc>
   0x0000052d <+31>:    add    $0x8,%esp
   0x00000530 <+34>:    mov    %eax,-0xc(%ebp)
   0x00000533 <+37>:    mov    $0x0,%eax
   0x00000538 <+42>:    leave
   0x00000539 <+43>:    ret
End of assembler dump.
```


ESP->
6
EBP
RET_ADDR
5
6
<- EBP
6
5
EBP
RET_ADDR
argc
argv
...
int
calc (int lhs,
int rhs)
{
int local1 = 1 + lhs;
int local2 = rhs - 2;
return local1 * local2;
}
int
main (int argc,
char **argv)
{
int a = 5;
int b = 6;
int r = calc (a, b);
return 0;
}
Dump of assembler code for function calc:
0x000004ed <+0>: push %ebp
0x000004ee <+1>: mov %esp,%ebp
0x000004f0 <+3>: sub $0x10,%esp
0x000004f3 <+6>: mov 0x8(%ebp),%eax
0x000004f6 <+9>: add $0x1,%eax
0x000004f9 <+12>: mov %eax,-0x4(%ebp)
0x000004fc <+15>: mov 0xc(%ebp),%eax
0x000004ff <+18>: sub $0x2,%eax
0x00000502 <+21>: mov %eax,-0x8(%ebp)
0x00000505 <+24>: mov -0x4(%ebp),%eax
0x00000508 <+27>: imul -0x8(%ebp),%eax
0x0000050c <+31>: leave
0x0000050d <+32>: ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
0x0000050e <+0>: push %ebp
0x0000050f <+1>: mov %esp,%ebp
0x00000511 <+3>: sub $0x10,%esp
0x00000514 <+6>: movl $0x5,-0x4(%ebp)
0x0000051b <+13>: movl $0x6,-0x8(%ebp)
0x00000522 <+20>: pushl -0x8(%ebp)
0x00000525 <+23>: pushl -0x4(%ebp)
0x00000528 <+26>: call 0x4ed <calc>
0x0000052d <+31>: add $0x8,%esp
0x00000530 <+34>: mov %eax,-0xc(%ebp)
0x00000533 <+37>: mov $0x0,%eax
0x00000538 <+42>: leave
0x00000539 <+43>: ret
End of assembler dump.



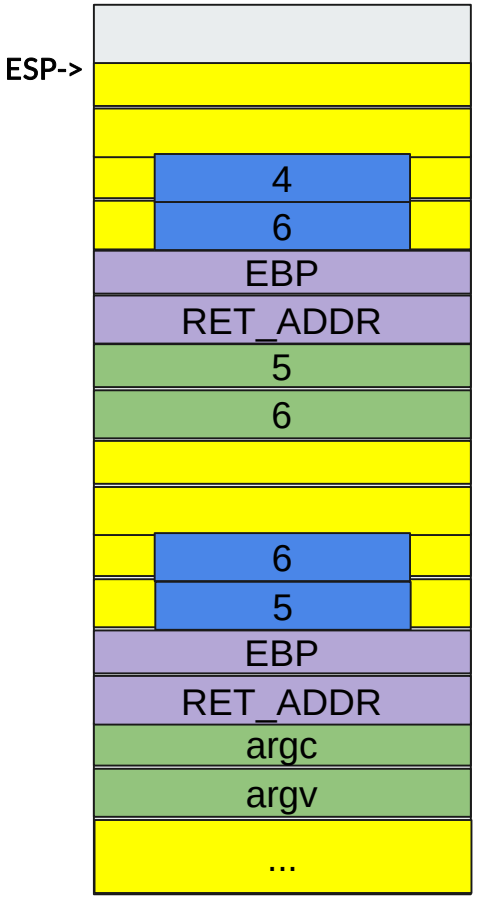

```
int
calc (int lhs,
      int rhs)
{
  int local1 = 1 + lhs;
  int local2 = rhs - 2;

  return local1 * local2;
}

int
main (int argc,
      char **argv)
{
  int a = 5;
  int b = 6;
  int r = calc (a, b);

  return 0;
}
```

```
Dump of assembler code for function calc:
   0x000004ed <+0>:     push   %ebp
   0x000004ee <+1>:     mov    %esp,%ebp
   0x000004f0 <+3>:     sub    $0x10,%esp
   0x000004f3 <+6>:     mov    0x8(%ebp),%eax
   0x000004f6 <+9>:     add    $0x1,%eax
   0x000004f9 <+12>:    mov    %eax,-0x4(%ebp)
   0x000004fc <+15>:    mov    0xc(%ebp),%eax
   0x000004ff <+18>:    sub    $0x2,%eax
   0x00000502 <+21>:    mov    %eax,-0x8(%ebp)
   0x00000505 <+24>:    mov    -0x4(%ebp),%eax
   0x00000508 <+27>:    imul   -0x8(%ebp),%eax
   0x0000050c <+31>:    leave
   0x0000050d <+32>:    ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
   0x0000050e <+0>:     push   %ebp
   0x0000050f <+1>:     mov    %esp,%ebp
   0x00000511 <+3>:     sub    $0x10,%esp
   0x00000514 <+6>:     movl   $0x5,-0x4(%ebp)
   0x0000051b <+13>:    movl   $0x6,-0x8(%ebp)
   0x00000522 <+20>:    pushl  -0x8(%ebp)
   0x00000525 <+23>:    pushl  -0x4(%ebp)
   0x00000528 <+26>:    call   0x4ed <calc>
   0x0000052d <+31>:    add    $0x8,%esp
   0x00000530 <+34>:    mov    %eax,-0xc(%ebp)
   0x00000533 <+37>:    mov    $0x0,%eax
   0x00000538 <+42>:    leave
   0x00000539 <+43>:    ret
End of assembler dump.
```



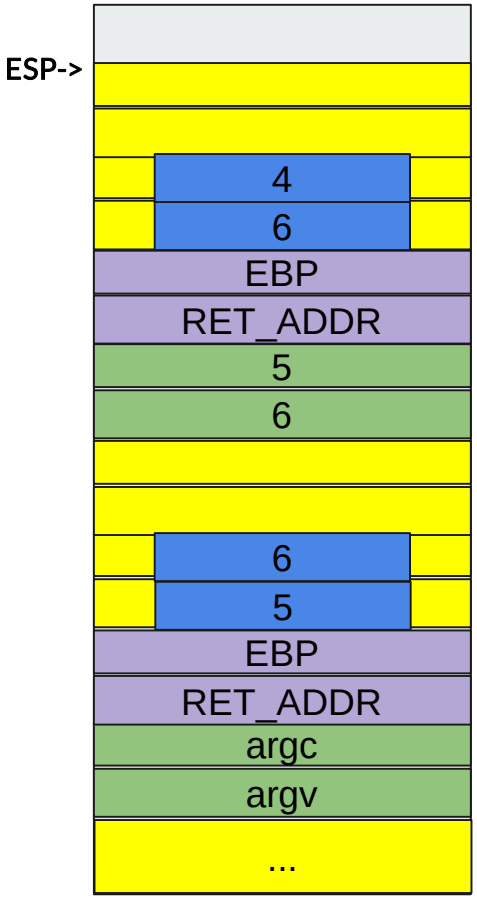

```
int
calc (int lhs,
      int rhs)
{
  int local1 = 1 + lhs;
  int local2 = rhs - 2;

  return local1 * local2;
}

int
main (int argc,
      char **argv)
{
  int a = 5;
  int b = 6;
  int r = calc (a, b);

  return 0;
}
```

```
Dump of assembler code for function calc:
   0x000004ed <+0>:     push   %ebp
   0x000004ee <+1>:     mov    %esp,%ebp
   0x000004f0 <+3>:     sub    $0x10,%esp
   0x000004f3 <+6>:     mov    0x8(%ebp),%eax
   0x000004f6 <+9>:     add    $0x1,%eax
   0x000004f9 <+12>:    mov    %eax,-0x4(%ebp)
   0x000004fc <+15>:    mov    0xc(%ebp),%eax
   0x000004ff <+18>:    sub    $0x2,%eax
   0x00000502 <+21>:    mov    %eax,-0x8(%ebp)
   0x00000505 <+24>:    mov    -0x4(%ebp),%eax
   0x00000508 <+27>:    imul   -0x8(%ebp),%eax
   0x0000050c <+31>:    leave
   0x0000050d <+32>:    ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
   0x0000050e <+0>:     push   %ebp
   0x0000050f <+1>:     mov    %esp,%ebp
   0x00000511 <+3>:     sub    $0x10,%esp
   0x00000514 <+6>:     movl   $0x5,-0x4(%ebp)
   0x0000051b <+13>:    movl   $0x6,-0x8(%ebp)
   0x00000522 <+20>:    pushl  -0x8(%ebp)
   0x00000525 <+23>:    pushl  -0x4(%ebp)
   0x00000528 <+26>:    call   0x4ed <calc>
   0x0000052d <+31>:    add    $0x8,%esp
   0x00000530 <+34>:    mov    %eax,-0xc(%ebp)
   0x00000533 <+37>:    mov    $0x0,%eax
   0x00000538 <+42>:    leave
   0x00000539 <+43>:    ret
End of assembler dump.
```



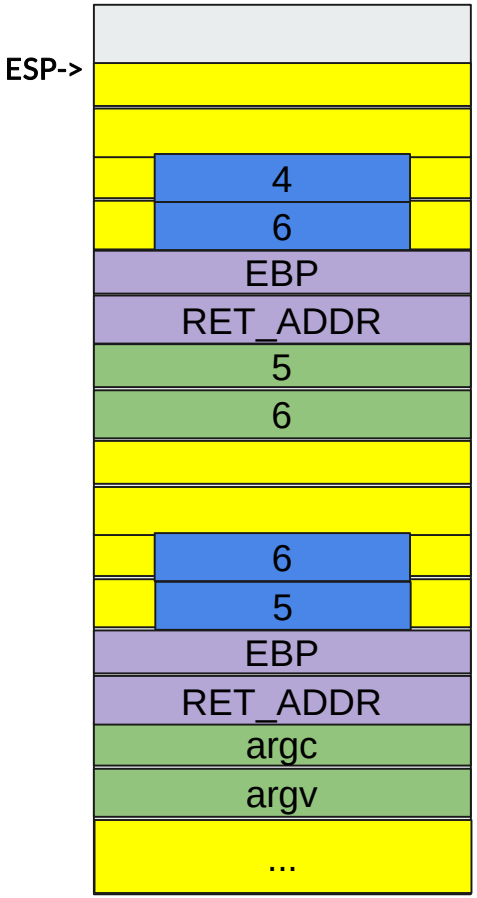

```
int
calc (int lhs,
      int rhs)
{
  int local1 = 1 + lhs;
  int local2 = rhs - 2;

  return local1 * local2;
}

int
main (int argc,
      char **argv)
{
  int a = 5;
  int b = 6;
  int r = calc (a, b);

  return 0;
}
```

```
Dump of assembler code for function calc:
   0x000004ed <+0>:     push   %ebp
   0x000004ee <+1>:     mov    %esp,%ebp
   0x000004f0 <+3>:     sub    $0x10,%esp
   0x000004f3 <+6>:     mov    0x8(%ebp),%eax
   0x000004f6 <+9>:     add    $0x1,%eax
   0x000004f9 <+12>:    mov    %eax,-0x4(%ebp)
   0x000004fc <+15>:    mov    0xc(%ebp),%eax
   0x000004ff <+18>:    sub    $0x2,%eax
   0x00000502 <+21>:    mov    %eax,-0x8(%ebp)
   0x00000505 <+24>:    mov    -0x4(%ebp),%eax
   0x00000508 <+27>:    imul   -0x8(%ebp),%eax
   0x0000050c <+31>:    leave
   0x0000050d <+32>:    ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
   0x0000050e <+0>:     push   %ebp
   0x0000050f <+1>:     mov    %esp,%ebp
   0x00000511 <+3>:     sub    $0x10,%esp
   0x00000514 <+6>:     movl   $0x5,-0x4(%ebp)
   0x0000051b <+13>:    movl   $0x6,-0x8(%ebp)
   0x00000522 <+20>:    pushl  -0x8(%ebp)
   0x00000525 <+23>:    pushl  -0x4(%ebp)
   0x00000528 <+26>:    call   0x4ed <calc>
   0x0000052d <+31>:    add    $0x8,%esp
   0x00000530 <+34>:    mov    %eax,-0xc(%ebp)
   0x00000533 <+37>:    mov    $0x0,%eax
   0x00000538 <+42>:    leave
   0x00000539 <+43>:    ret
End of assembler dump.
```



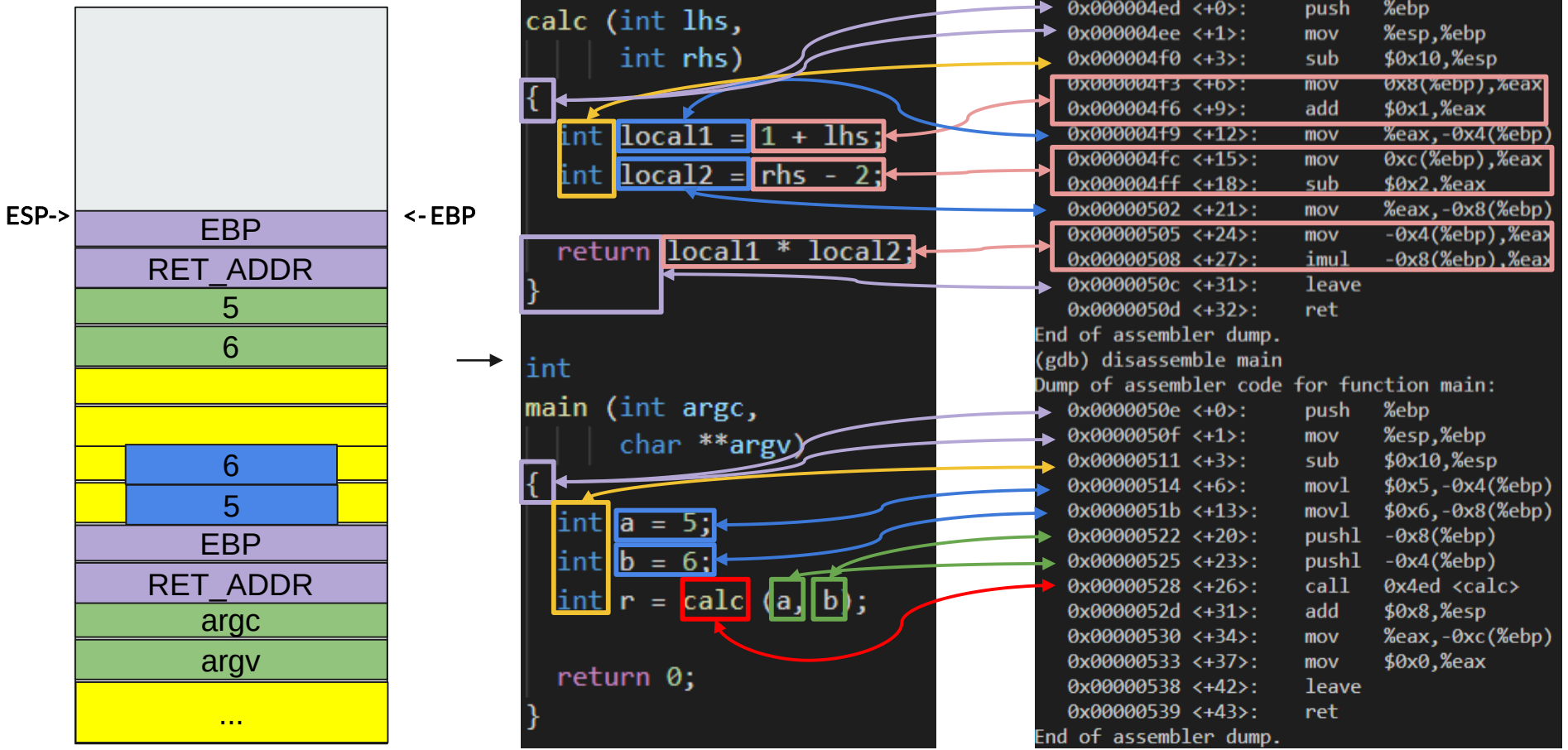

```
int
calc (int lhs,
      int rhs)
{
  int local1 = 1 + lhs;
  int local2 = rhs - 2;

  return local1 * local2;
}

int
main (int argc,
      char **argv)
{
  int a = 5;
  int b = 6;
  int r = calc (a, b);

  return 0;
}
```

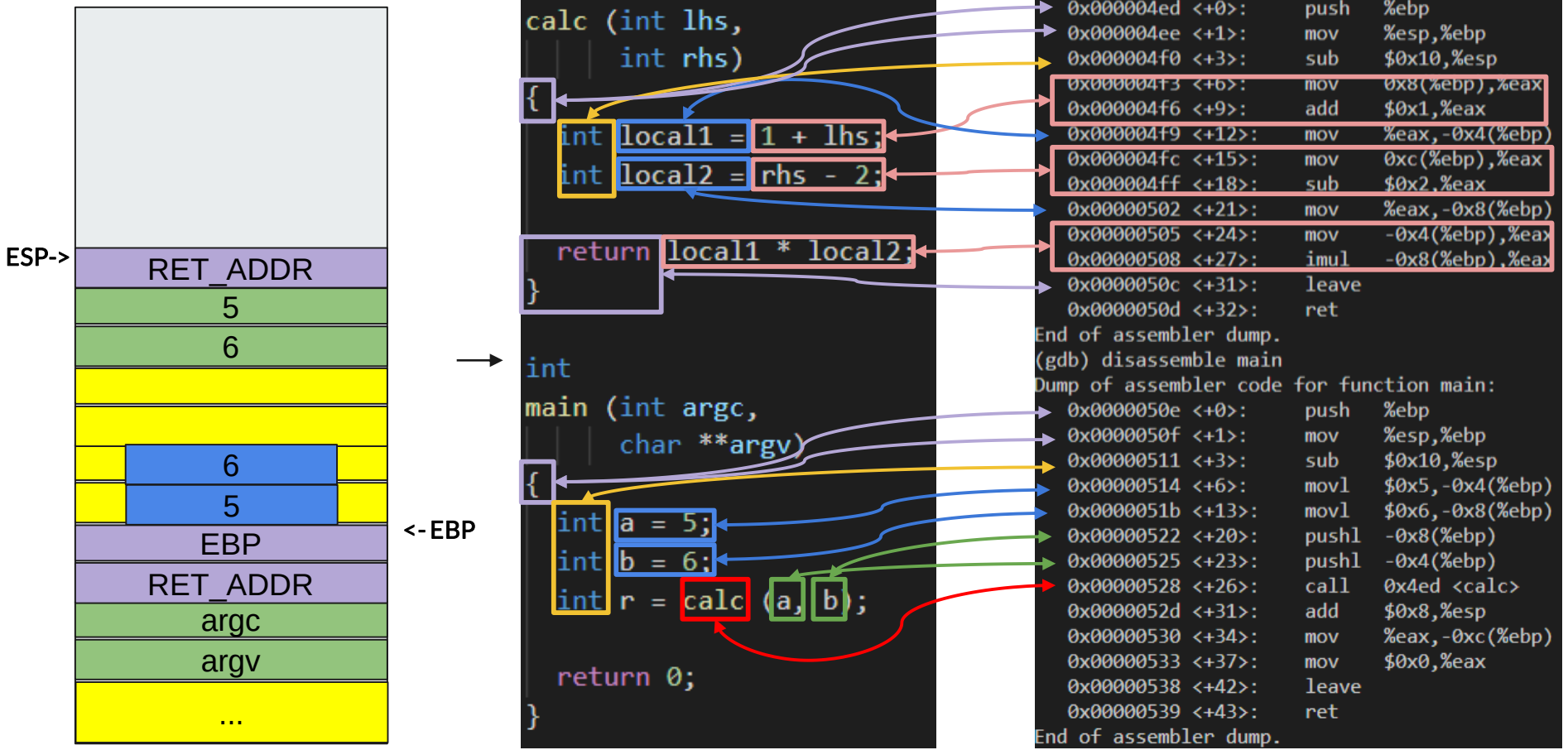
```
Dump of assembler code for function calc:
   0x000004ed <+0>:     push   %ebp
   0x000004ee <+1>:     mov    %esp,%ebp
   0x000004f0 <+3>:     sub    $0x10,%esp
   0x000004f3 <+6>:     mov    0x8(%ebp),%eax
   0x000004f6 <+9>:     add    $0x1,%eax
   0x000004f9 <+12>:    mov    %eax,-0x4(%ebp)
   0x000004fc <+15>:    mov    0xc(%ebp),%eax
   0x000004ff <+18>:    sub    $0x2,%eax
   0x00000502 <+21>:    mov    %eax,-0x8(%ebp)
   0x00000505 <+24>:    mov    -0x4(%ebp),%eax
   0x00000508 <+27>:    imul   -0x8(%ebp),%eax
   0x0000050c <+31>:    leave
   0x0000050d <+32>:    ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
   0x0000050e <+0>:     push   %ebp
   0x0000050f <+1>:     mov    %esp,%ebp
   0x00000511 <+3>:     sub    $0x10,%esp
   0x00000514 <+6>:     movl   $0x5,-0x4(%ebp)
   0x0000051b <+13>:    movl   $0x6,-0x8(%ebp)
   0x00000522 <+20>:    pushl  -0x8(%ebp)
   0x00000525 <+23>:    pushl  -0x4(%ebp)
   0x00000528 <+26>:    call   0x4ed <calc>
   0x0000052d <+31>:    add    $0x8,%esp
   0x00000530 <+34>:    mov    %eax,-0xc(%ebp)
   0x00000533 <+37>:    mov    $0x0,%eax
   0x00000538 <+42>:    leave
   0x00000539 <+43>:    ret
End of assembler dump.
```

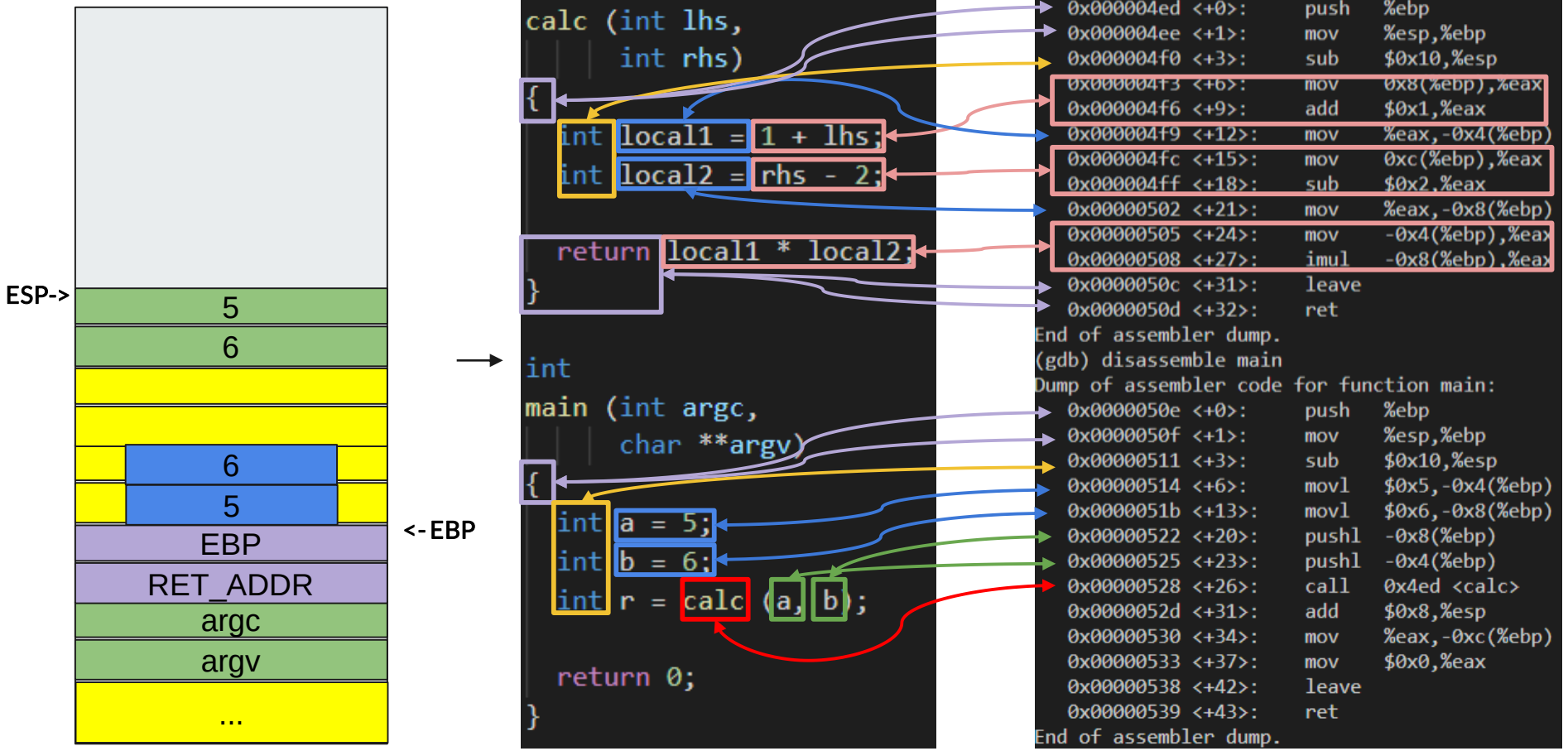


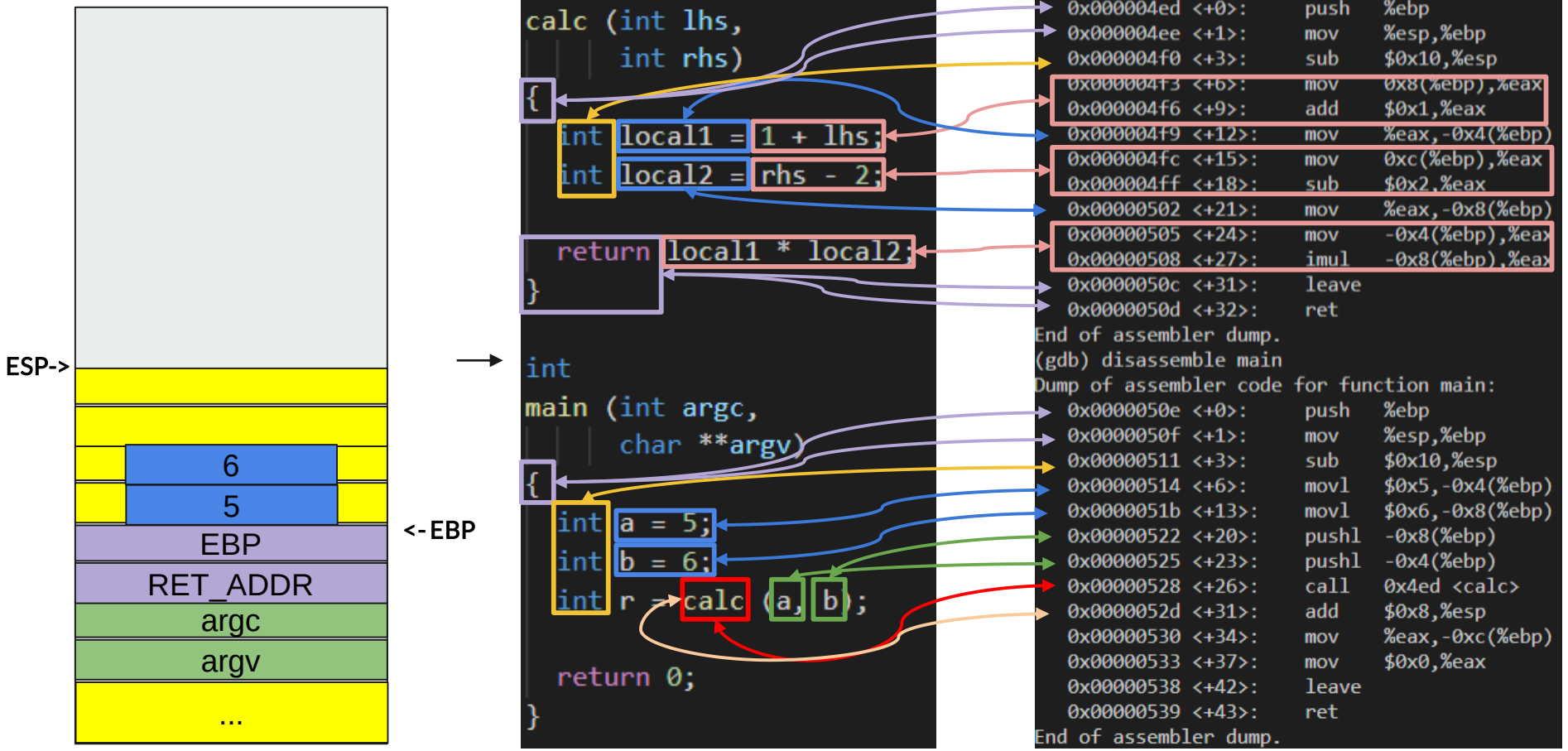

```
int
calc (int lhs,
      int rhs)
{
  int local1 = 1 + lhs;
  int local2 = rhs - 2;

  return local1 * local2;
}

int
main (int argc,
      char **argv)
{
  int a = 5;
  int b = 6;
  int r = calc (a, b);

  return 0;
}
```

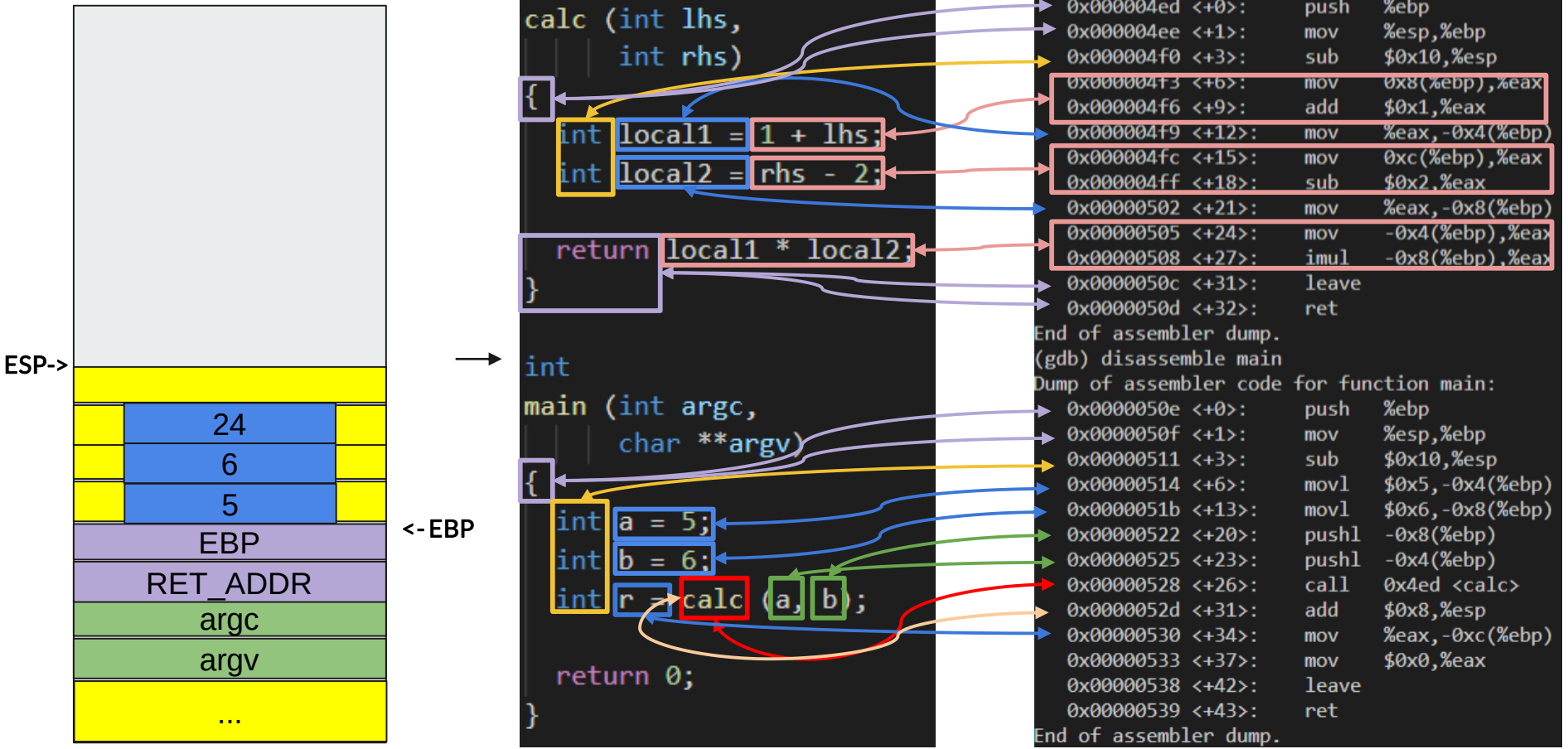
```
Dump of assembler code for function calc:
   0x000004ed <+0>:     push   %ebp
   0x000004ee <+1>:     mov    %esp,%ebp
   0x000004f0 <+3>:     sub    $0x10,%esp
   0x000004f3 <+6>:     mov    0x8(%ebp),%eax
   0x000004f6 <+9>:     add    $0x1,%eax
   0x000004f9 <+12>:    mov    %eax,-0x4(%ebp)
   0x000004fc <+15>:    mov    0xc(%ebp),%eax
   0x000004ff <+18>:    sub    $0x2,%eax
   0x00000502 <+21>:    mov    %eax,-0x8(%ebp)
   0x00000505 <+24>:    mov    -0x4(%ebp),%eax
   0x00000508 <+27>:    imul   -0x8(%ebp),%eax
   0x0000050c <+31>:    leave
   0x0000050d <+32>:    ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
   0x0000050e <+0>:     push   %ebp
   0x0000050f <+1>:     mov    %esp,%ebp
   0x00000511 <+3>:     sub    $0x10,%esp
   0x00000514 <+6>:     movl   $0x5,-0x4(%ebp)
   0x0000051b <+13>:    movl   $0x6,-0x8(%ebp)
   0x00000522 <+20>:    pushl  -0x8(%ebp)
   0x00000525 <+23>:    pushl  -0x4(%ebp)
   0x00000528 <+26>:    call   0x4ed <calc>
   0x0000052d <+31>:    add    $0x8,%esp
   0x00000530 <+34>:    mov    %eax,-0xc(%ebp)
   0x00000533 <+37>:    mov    $0x0,%eax
   0x00000538 <+42>:    leave
   0x00000539 <+43>:    ret
End of assembler dump.
```

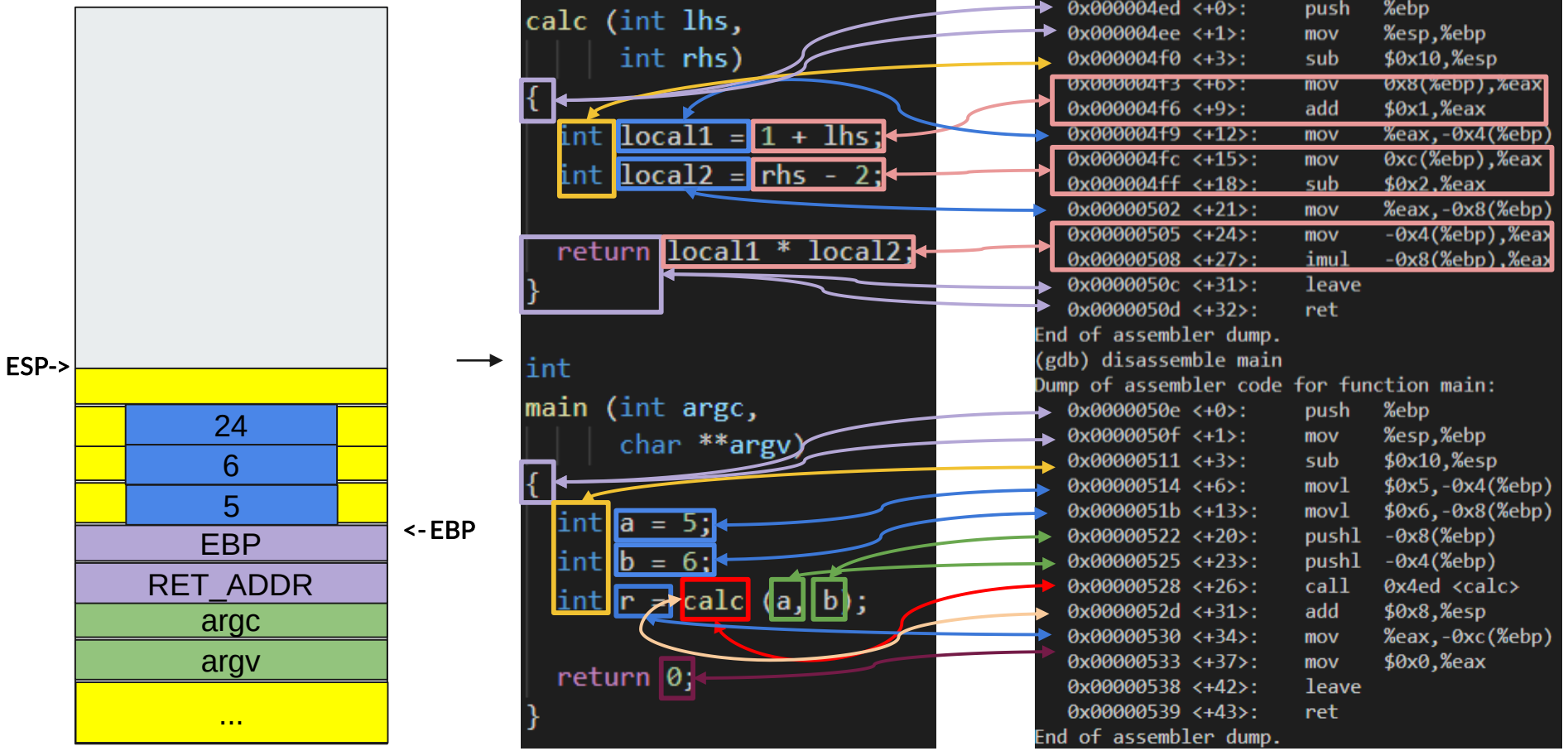

ESP->
EBP
<- EBP
RET_ADDR
5
6
6
5
EBP
RET_ADDR
argc
argv
...
int
calc (int lhs,
int rhs)
{
int local1 = 1 + lhs;
int local2 = rhs - 2;
return local1 * local2;
}
int
main (int argc,
char **argv)
{
int a = 5;
int b = 6;
int r = calc (a, b);
return 0;
}
Dump of assembler code for function calc:
0x000004ed <+0>: push %ebp
0x000004ee <+1>: mov %esp,%ebp
0x000004f0 <+3>: sub $0x10,%esp
0x000004f3 <+6>: mov 0x8(%ebp),%eax
0x000004f6 <+9>: add $0x1,%eax
0x000004f9 <+12>: mov %eax,-0x4(%ebp)
0x000004fc <+15>: mov 0xc(%ebp),%eax
0x000004ff <+18>: sub $0x2,%eax
0x00000502 <+21>: mov %eax,-0x8(%ebp)
0x00000505 <+24>: mov -0x4(%ebp),%eax
0x00000508 <+27>: imul -0x8(%ebp),%eax
0x0000050c <+31>: leave
0x0000050d <+32>: ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
0x0000050e <+0>: push %ebp
0x0000050f <+1>: mov %esp,%ebp
0x00000511 <+3>: sub $0x10,%esp
0x00000514 <+6>: movl $0x5,-0x4(%ebp)
0x0000051b <+13>: movl $0x6,-0x8(%ebp)
0x00000522 <+20>: pushl -0x8(%ebp)
0x00000525 <+23>: pushl -0x4(%ebp)
0x00000528 <+26>: call 0x4ed <calc>
0x0000052d <+31>: add $0x8,%esp
0x00000530 <+34>: mov %eax,-0xc(%ebp)
0x00000533 <+37>: mov $0x0,%eax
0x00000538 <+42>: leave
0x00000539 <+43>: ret
End of assembler dump.


ESP->
RET_ADDR
5
6
6
5
EBP
<- EBP
RET_ADDR
argc
argv
...
int
calc (int lhs,
int rhs)
{
int local1 = 1 + lhs;
int local2 = rhs - 2;
return local1 * local2;
}
int
main (int argc,
char **argv)
{
int a = 5;
int b = 6;
int r = calc (a, b);
return 0;
}
Dump of assembler code for function calc:
0x000004ed <+0>: push %ebp
0x000004ee <+1>: mov %esp,%ebp
0x000004f0 <+3>: sub $0x10,%esp
0x000004f3 <+6>: mov 0x8(%ebp),%eax
0x000004f6 <+9>: add $0x1,%eax
0x000004f9 <+12>: mov %eax,-0x4(%ebp)
0x000004fc <+15>: mov 0xc(%ebp),%eax
0x000004ff <+18>: sub $0x2,%eax
0x00000502 <+21>: mov %eax,-0x8(%ebp)
0x00000505 <+24>: mov -0x4(%ebp),%eax
0x00000508 <+27>: imul -0x8(%ebp),%eax
0x0000050c <+31>: leave
0x0000050d <+32>: ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
0x0000050e <+0>: push %ebp
0x0000050f <+1>: mov %esp,%ebp
0x00000511 <+3>: sub $0x10,%esp
0x00000514 <+6>: movl $0x5,-0x4(%ebp)
0x0000051b <+13>: movl $0x6,-0x8(%ebp)
0x00000522 <+20>: pushl -0x8(%ebp)
0x00000525 <+23>: pushl -0x4(%ebp)
0x00000528 <+26>: call 0x4ed <calc>
0x0000052d <+31>: add $0x8,%esp
0x00000530 <+34>: mov %eax,-0xc(%ebp)
0x00000533 <+37>: mov $0x0,%eax
0x00000538 <+42>: leave
0x00000539 <+43>: ret
End of assembler dump.

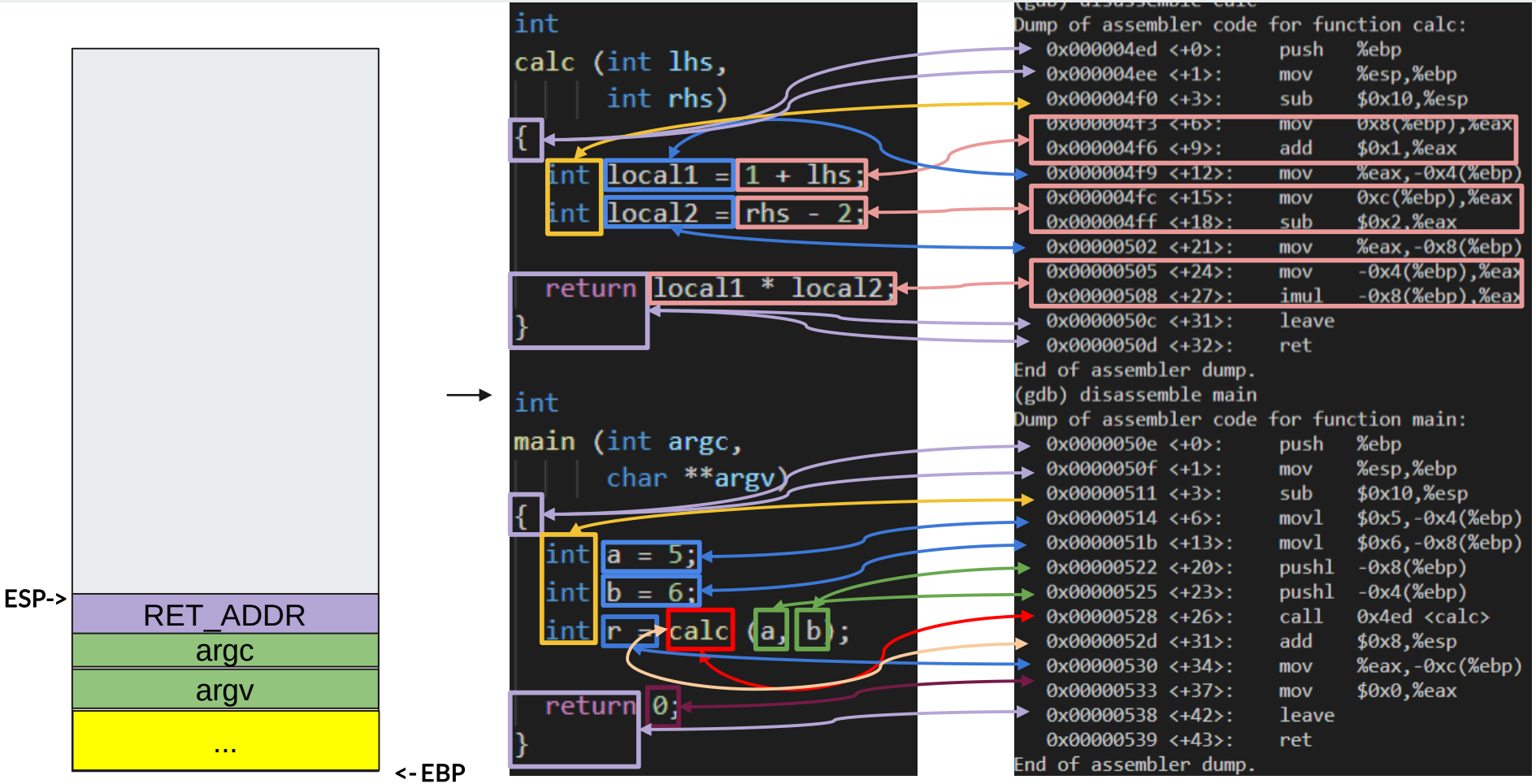

ESP->
5
6
6
5
EBP
RET_ADDR
argc
argv
...
<-EBP
int
calc (int lhs,
int rhs)
{
int local1 = 1 + lhs;
int local2 = rhs - 2;
return local1 * local2;
}
int
main (int argc,
char **argv)
{
int a = 5;
int b = 6;
int r = calc (a, b);
return 0;
}
Dump of assembler code for function calc:
0x000004ed <+0>: push %ebp
0x000004ee <+1>: mov %esp,%ebp
0x000004f0 <+3>: sub $0x10,%esp
0x000004f3 <+6>: mov 0x8(%ebp),%eax
0x000004f6 <+9>: add $0x1,%eax
0x000004f9 <+12>: mov %eax,-0x4(%ebp)
0x000004fc <+15>: mov 0xc(%ebp),%eax
0x000004ff <+18>: sub $0x2,%eax
0x00000502 <+21>: mov %eax,-0x8(%ebp)
0x00000505 <+24>: mov -0x4(%ebp),%eax
0x00000508 <+27>: imul -0x8(%ebp),%eax
0x0000050c <+31>: leave
0x0000050d <+32>: ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
0x0000050e <+0>: push %ebp
0x0000050f <+1>: mov %esp,%ebp
0x00000511 <+3>: sub $0x10,%esp
0x00000514 <+6>: movl $0x5,-0x4(%ebp)
0x0000051b <+13>: movl $0x6,-0x8(%ebp)
0x00000522 <+20>: pushl -0x8(%ebp)
0x00000525 <+23>: pushl -0x4(%ebp)
0x00000528 <+26>: call 0x4ed <calc>
0x0000052d <+31>: add $0x8,%esp
0x00000530 <+34>: mov %eax,-0xc(%ebp)
0x00000533 <+37>: mov $0x0,%eax
0x00000538 <+42>: leave
0x00000539 <+43>: ret
End of assembler dump.

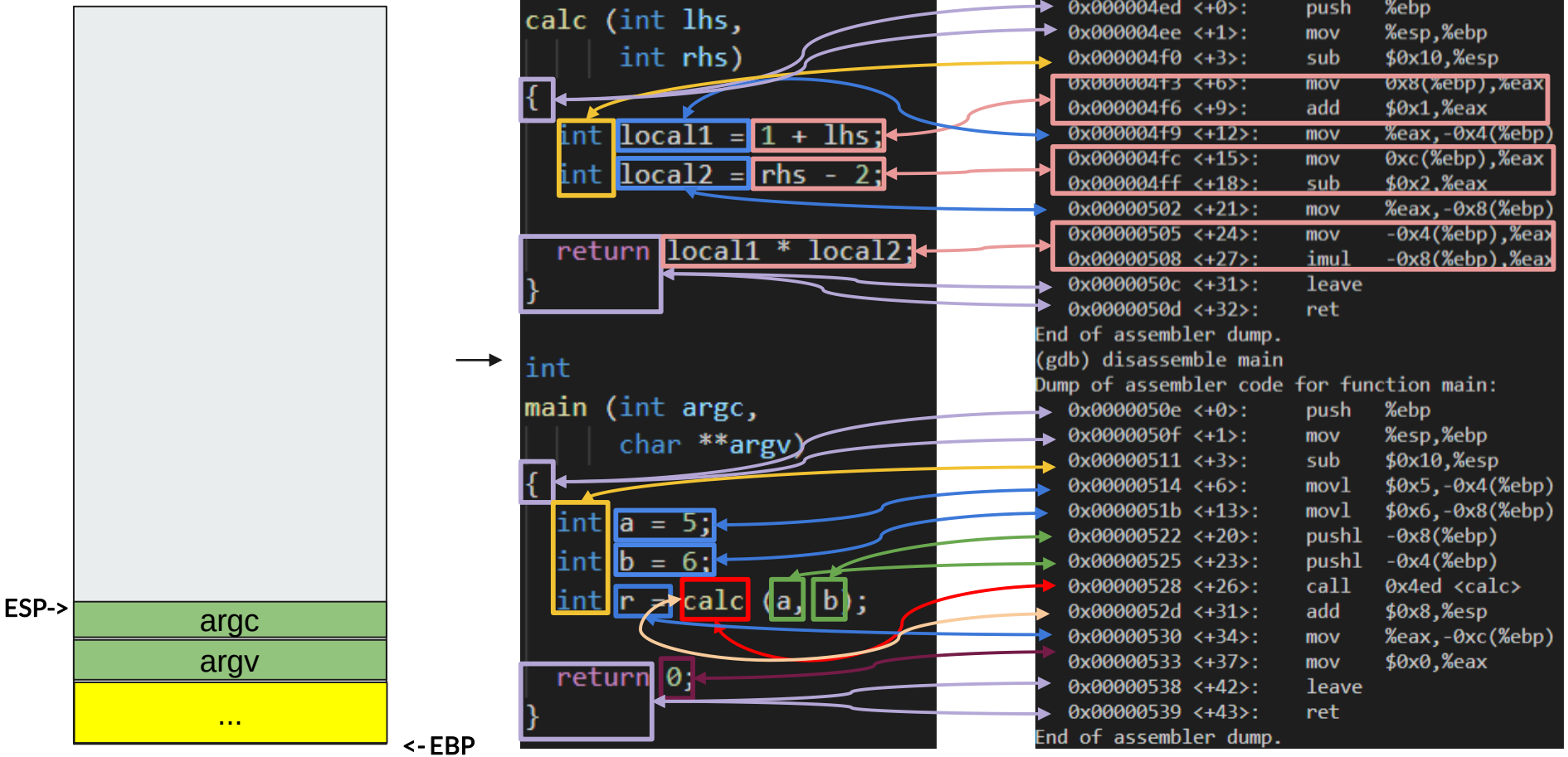

ESP->
6
5
EBP
<- EBP
RET_ADDR
argc
argv
...
int
calc (int lhs,
int rhs)
{
int local1 = 1 + lhs;
int local2 = rhs - 2;
return local1 * local2;
}
int
main (int argc,
char **argv)
{
int a = 5;
int b = 6;
int r = calc (a, b);
return 0;
}
Dump of assembler code for function calc:
0x000004ed <+0>: push %ebp
0x000004ee <+1>: mov %esp,%ebp
0x000004f0 <+3>: sub $0x10,%esp
0x000004f3 <+6>: mov 0x8(%ebp),%eax
0x000004f6 <+9>: add $0x1,%eax
0x000004f9 <+12>: mov %eax,-0x4(%ebp)
0x000004fc <+15>: mov 0xc(%ebp),%eax
0x000004ff <+18>: sub $0x2,%eax
0x00000502 <+21>: mov %eax,-0x8(%ebp)
0x00000505 <+24>: mov -0x4(%ebp),%eax
0x00000508 <+27>: imul -0x8(%ebp),%eax
0x0000050c <+31>: leave
0x0000050d <+32>: ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
0x0000050e <+0>: push %ebp
0x0000050f <+1>: mov %esp,%ebp
0x00000511 <+3>: sub $0x10,%esp
0x00000514 <+6>: movl $0x5,-0x4(%ebp)
0x0000051b <+13>: movl $0x6,-0x8(%ebp)
0x00000522 <+20>: pushl -0x8(%ebp)
0x00000525 <+23>: pushl -0x4(%ebp)
0x00000528 <+26>: call 0x4ed <calc>
0x0000052d <+31>: add $0x8,%esp
0x00000530 <+34>: mov %eax,-0xc(%ebp)
0x00000533 <+37>: mov $0x0,%eax
0x00000538 <+42>: leave
0x00000539 <+43>: ret
End of assembler dump.

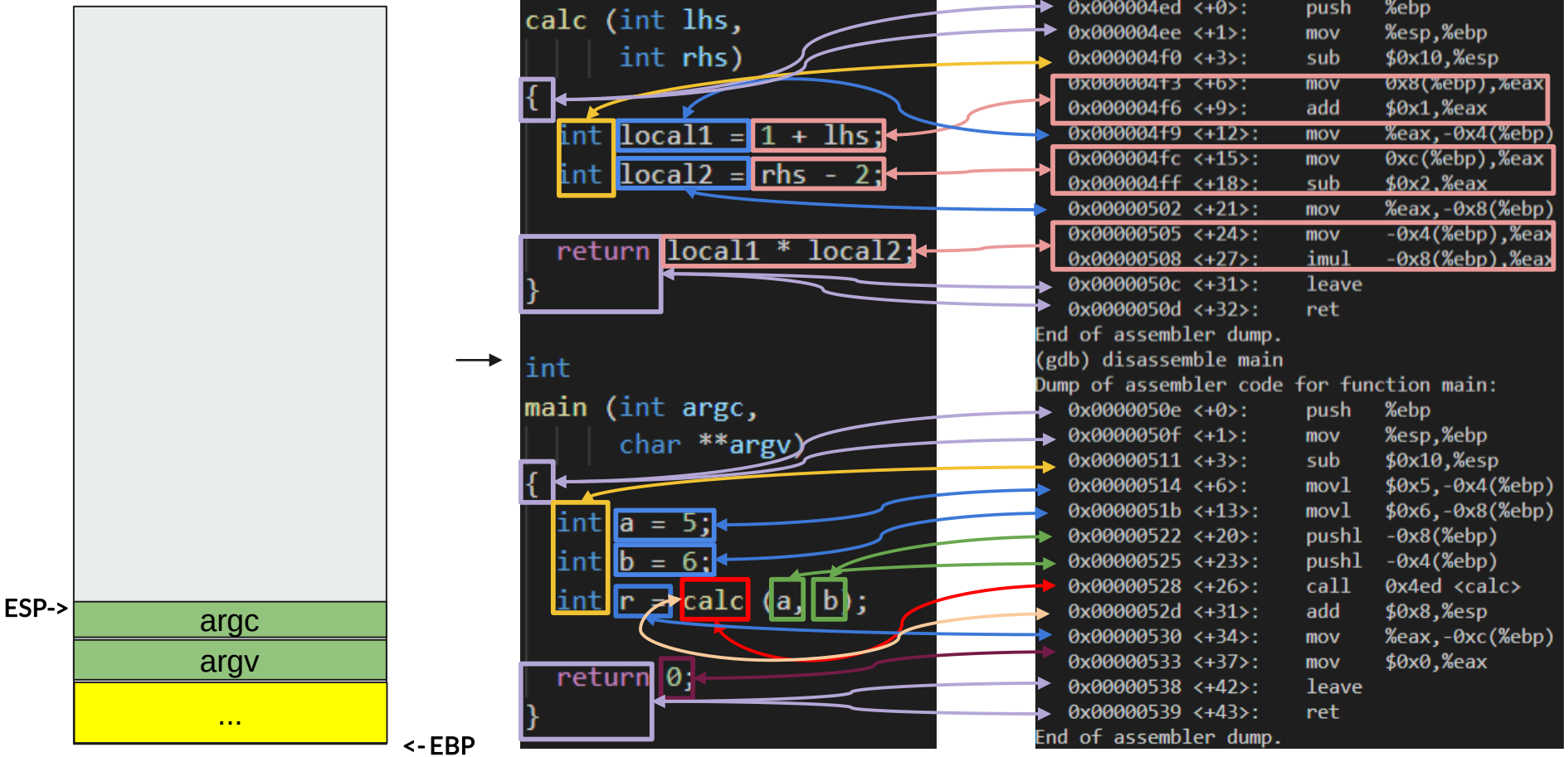

ESP->
24
6
5
EBP
<- EBP
RET_ADDR
argc
argv
...
int
calc (int lhs,
int rhs)
{
int local1 = 1 + lhs;
int local2 = rhs - 2;
return local1 * local2;
}
int
main (int argc,
char **argv)
{
int a = 5;
int b = 6;
int r = calc (a, b);
return 0;
}
Dump of assembler code for function calc:
0x000004ed <+0>: push %ebp
0x000004ee <+1>: mov %esp,%ebp
0x000004f0 <+3>: sub $0x10,%esp
0x000004f3 <+6>: mov 0x8(%ebp),%eax
0x000004f6 <+9>: add $0x1,%eax
0x000004f9 <+12>: mov %eax,-0x4(%ebp)
0x000004fc <+15>: mov 0xc(%ebp),%eax
0x000004ff <+18>: sub $0x2,%eax
0x00000502 <+21>: mov %eax,-0x8(%ebp)
0x00000505 <+24>: mov -0x4(%ebp),%eax
0x00000508 <+27>: imul -0x8(%ebp),%eax
0x0000050c <+31>: leave
0x0000050d <+32>: ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
0x0000050e <+0>: push %ebp
0x0000050f <+1>: mov %esp,%ebp
0x00000511 <+3>: sub $0x10,%esp
0x00000514 <+6>: movl $0x5,-0x4(%ebp)
0x0000051b <+13>: movl $0x6,-0x8(%ebp)
0x00000522 <+20>: pushl -0x8(%ebp)
0x00000525 <+23>: pushl -0x4(%ebp)
0x00000528 <+26>: call 0x4ed <calc>
0x0000052d <+31>: add $0x8,%esp
0x00000530 <+34>: mov %eax,-0xc(%ebp)
0x00000533 <+37>: mov $0x0,%eax
0x00000538 <+42>: leave
0x00000539 <+43>: ret
End of assembler dump.


ESP->
24
6
5
EBP
<- EBP
RET_ADDR
argc
argv
...
int
calc (int lhs,
int rhs)
{
int local1 = 1 + lhs;
int local2 = rhs - 2;
return local1 * local2;
}
int
main (int argc,
char **argv)
{
int a = 5;
int b = 6;
int r = calc (a, b);
return 0;
}
Dump of assembler code for function calc:
0x000004ed <+0>: push %ebp
0x000004ee <+1>: mov %esp,%ebp
0x000004f0 <+3>: sub $0x10,%esp
0x000004f3 <+6>: mov 0x8(%ebp),%eax
0x000004f6 <+9>: add $0x1,%eax
0x000004f9 <+12>: mov %eax,-0x4(%ebp)
0x000004fc <+15>: mov 0xc(%ebp),%eax
0x000004ff <+18>: sub $0x2,%eax
0x00000502 <+21>: mov %eax,-0x8(%ebp)
0x00000505 <+24>: mov -0x4(%ebp),%eax
0x00000508 <+27>: imul -0x8(%ebp),%eax
0x0000050c <+31>: leave
0x0000050d <+32>: ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
0x0000050e <+0>: push %ebp
0x0000050f <+1>: mov %esp,%ebp
0x00000511 <+3>: sub $0x10,%esp
0x00000514 <+6>: movl $0x5,-0x4(%ebp)
0x0000051b <+13>: movl $0x6,-0x8(%ebp)
0x00000522 <+20>: pushl -0x8(%ebp)
0x00000525 <+23>: pushl -0x4(%ebp)
0x00000528 <+26>: call 0x4ed <calc>
0x0000052d <+31>: add $0x8,%esp
0x00000530 <+34>: mov %eax,-0xc(%ebp)
0x00000533 <+37>: mov $0x0,%eax
0x00000538 <+42>: leave
0x00000539 <+43>: ret
End of assembler dump.


ESP->
EBP
<- EBP
RET_ADDR
argc
argv
...
int
calc (int lhs,
int rhs)
{
int local1 = 1 + lhs;
int local2 = rhs - 2;
return local1 * local2;
}
int
main (int argc,
char **argv)
{
int a = 5;
int b = 6;
int r = calc (a, b);
return 0;
}
Dump of assembler code for function calc:
0x000004ed <+0>: push %ebp
0x000004ee <+1>: mov %esp,%ebp
0x000004f0 <+3>: sub $0x10,%esp
0x000004f3 <+6>: mov 0x8(%ebp),%eax
0x000004f6 <+9>: add $0x1,%eax
0x000004f9 <+12>: mov %eax,-0x4(%ebp)
0x000004fc <+15>: mov 0xc(%ebp),%eax
0x000004ff <+18>: sub $0x2,%eax
0x00000502 <+21>: mov %eax,-0x8(%ebp)
0x00000505 <+24>: mov -0x4(%ebp),%eax
0x00000508 <+27>: imul -0x8(%ebp),%eax
0x0000050c <+31>: leave
0x0000050d <+32>: ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
0x0000050e <+0>: push %ebp
0x0000050f <+1>: mov %esp,%ebp
0x00000511 <+3>: sub $0x10,%esp
0x00000514 <+6>: movl $0x5,-0x4(%ebp)
0x0000051b <+13>: movl $0x6,-0x8(%ebp)
0x00000522 <+20>: pushl -0x8(%ebp)
0x00000525 <+23>: pushl -0x4(%ebp)
0x00000528 <+26>: call 0x4ed <calc>
0x0000052d <+31>: add $0x8,%esp
0x00000530 <+34>: mov %eax,-0xc(%ebp)
0x00000533 <+37>: mov $0x0,%eax
0x00000538 <+42>: leave
0x00000539 <+43>: ret
End of assembler dump.


ESP->
RET_ADDR
argc
argv
...
<- EBP
int
calc (int lhs,
int rhs)
{
int local1 = 1 + lhs;
int local2 = rhs - 2;
return local1 * local2;
}
int
main (int argc,
char **argv)
{
int a = 5;
int b = 6;
int r = calc (a, b);
return 0;
}
Dump of assembler code for function calc:
0x000004ed <+0>: push %ebp
0x000004ee <+1>: mov %esp,%ebp
0x000004f0 <+3>: sub $0x10,%esp
0x000004f3 <+6>: mov 0x8(%ebp),%eax
0x000004f6 <+9>: add $0x1,%eax
0x000004f9 <+12>: mov %eax,-0x4(%ebp)
0x000004fc <+15>: mov 0xc(%ebp),%eax
0x000004ff <+18>: sub $0x2,%eax
0x00000502 <+21>: mov %eax,-0x8(%ebp)
0x00000505 <+24>: mov -0x4(%ebp),%eax
0x00000508 <+27>: imul -0x8(%ebp),%eax
0x0000050c <+31>: leave
0x0000050d <+32>: ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
0x0000050e <+0>: push %ebp
0x0000050f <+1>: mov %esp,%ebp
0x00000511 <+3>: sub $0x10,%esp
0x00000514 <+6>: movl $0x5,-0x4(%ebp)
0x0000051b <+13>: movl $0x6,-0x8(%ebp)
0x00000522 <+20>: pushl -0x8(%ebp)
0x00000525 <+23>: pushl -0x4(%ebp)
0x00000528 <+26>: call 0x4ed <calc>
0x0000052d <+31>: add $0x8,%esp
0x00000530 <+34>: mov %eax,-0xc(%ebp)
0x00000533 <+37>: mov $0x0,%eax
0x00000538 <+42>: leave
0x00000539 <+43>: ret
End of assembler dump.


ESP->
argc
argv
...
<- EBP
int
calc (int lhs,
int rhs)
{
int local1 = 1 + lhs;
int local2 = rhs - 2;
return local1 * local2;
}
int
main (int argc,
char **argv)
{
int a = 5;
int b = 6;
int r = calc (a, b);
return 0;
}
Dump of assembler code for function calc:
0x000004ed <+0>: push %ebp
0x000004ee <+1>: mov %esp,%ebp
0x000004f0 <+3>: sub $0x10,%esp
0x000004f3 <+6>: mov 0x8(%ebp),%eax
0x000004f6 <+9>: add $0x1,%eax
0x000004f9 <+12>: mov %eax,-0x4(%ebp)
0x000004fc <+15>: mov 0xc(%ebp),%eax
0x000004ff <+18>: sub $0x2,%eax
0x00000502 <+21>: mov %eax,-0x8(%ebp)
0x00000505 <+24>: mov -0x4(%ebp),%eax
0x00000508 <+27>: imul -0x8(%ebp),%eax
0x0000050c <+31>: leave
0x0000050d <+32>: ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
0x0000050e <+0>: push %ebp
0x0000050f <+1>: mov %esp,%ebp
0x00000511 <+3>: sub $0x10,%esp
0x00000514 <+6>: movl $0x5,-0x4(%ebp)
0x0000051b <+13>: movl $0x6,-0x8(%ebp)
0x00000522 <+20>: pushl -0x8(%ebp)
0x00000525 <+23>: pushl -0x4(%ebp)
0x00000528 <+26>: call 0x4ed <calc>
0x0000052d <+31>: add $0x8,%esp
0x00000530 <+34>: mov %eax,-0xc(%ebp)
0x00000533 <+37>: mov $0x0,%eax
0x00000538 <+42>: leave
0x00000539 <+43>: ret
End of assembler dump.


ESP->
argc
argv
...
<- EBP
int
calc (int lhs,
int rhs)
{
int local1 = 1 + lhs;
int local2 = rhs - 2;
return local1 * local2;
}
int
main (int argc,
char **argv)
{
int a = 5;
int b = 6;
int r = calc (a, b);
return 0;
}
Dump of assembler code for function calc:
0x000004ed <+0>: push %ebp
0x000004ee <+1>: mov %esp,%ebp
0x000004f0 <+3>: sub $0x10,%esp
0x000004f3 <+6>: mov 0x8(%ebp),%eax
0x000004f6 <+9>: add $0x1,%eax
0x000004f9 <+12>: mov %eax,-0x4(%ebp)
0x000004fc <+15>: mov 0xc(%ebp),%eax
0x000004ff <+18>: sub $0x2,%eax
0x00000502 <+21>: mov %eax,-0x8(%ebp)
0x00000505 <+24>: mov -0x4(%ebp),%eax
0x00000508 <+27>: imul -0x8(%ebp),%eax
0x0000050c <+31>: leave
0x0000050d <+32>: ret
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
0x0000050e <+0>: push %ebp
0x0000050f <+1>: mov %esp,%ebp
0x00000511 <+3>: sub $0x10,%esp
0x00000514 <+6>: movl $0x5,-0x4(%ebp)
0x0000051b <+13>: movl $0x6,-0x8(%ebp)
0x00000522 <+20>: pushl -0x8(%ebp)
0x00000525 <+23>: pushl -0x4(%ebp)
0x00000528 <+26>: call 0x4ed <calc>
0x0000052d <+31>: add $0x8,%esp
0x00000530 <+34>: mov %eax,-0xc(%ebp)
0x00000533 <+37>: mov $0x0,%eax
0x00000538 <+42>: leave
0x00000539 <+43>: ret
End of assembler dump.