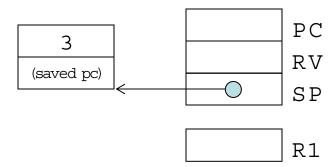
```
irt fact(irt n)
{
    if (n == 0) return 1;
    return n * fact(n - 1);
}
```

```
int fact (int n)
  if (n == 0) return 1;
  return n * fact(n - 1);
R1 = M[SP + 4];
BNE R1, 0, PC + 12;
RV = 1;
RET;
R1 = M[SP + 4];
R1 = R1 - 1;
SP = SP - 4;
M[SP] = R1;
CALL <fact>;
SP = SP + 4;
R1 = M[SP + 4];
RV = RV * R1;
RET;
```



```
int fact (int n)
  if (n == 0) return 1;
                                              3
  return n * fact(n - 1);
                                           (saved pc)
R1 = M[SP + 4];
BNE R1, 0, PC + 12;
RV = 1;
RET;
R1 = M[SP + 4];
R1 = R1 - 1;
SP = SP - 4;
M[SP] = R1;
CALL <fact>;
SP = SP + 4;
R1 = M[SP + 4];
```

RV = RV * R1;

RET;

PС

RV

SP

R1

```
int fact (int n)
  if (n == 0) return 1;
                                              3
  return n * fact(n - 1);
                                           (saved pc)
R1 = M[SP + 4];
BNE R1, 0, PC + 12;
RV = 1;
RET;
R1 = M[SP + 4];
R1 = R1 - 1;
SP = SP - 4;
M[SP] = R1;
CALL <fact>;
SP = SP + 4;
R1 = M[SP + 4];
```

RV = RV * R1;

RET;

3

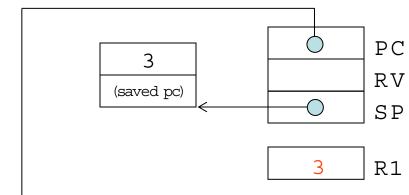
PС

RV

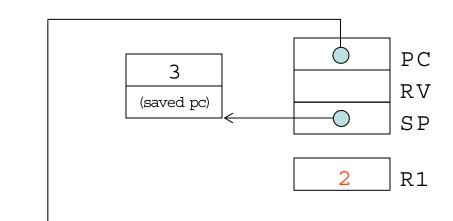
SP

R1

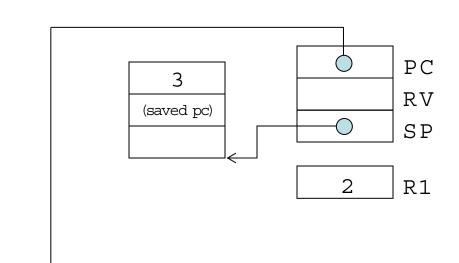
```
int fact (int n)
  if (n == 0) return 1;
  return n * fact(n - 1);
R1 = M[SP + 4];
BNE R1, 0, PC + 12;
RV = 1;
RET;
R1 = M[SP + 4];
R1 = R1 - 1;
SP = SP - 4;
M[SP] = R1;
CALL <fact>;
SP = SP + 4;
R1 = M[SP + 4];
RV = RV * R1;
RET;
```



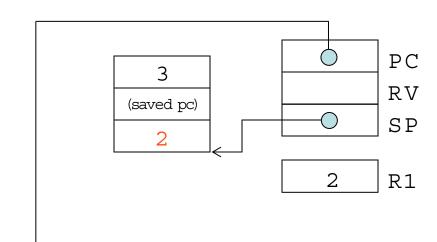
```
int fact (int n)
  if (n == 0) return 1;
  return n * fact(n - 1);
R1 = M[SP + 4];
BNE R1, 0, PC + 12;
RV = 1;
RET;
R1 = M[SP + 4];
R1 = R1 - 1;
SP = SP - 4;
M[SP] = R1;
CALL <fact>;
SP = SP + 4;
R1 = M[SP + 4];
RV = RV * R1;
RET;
```



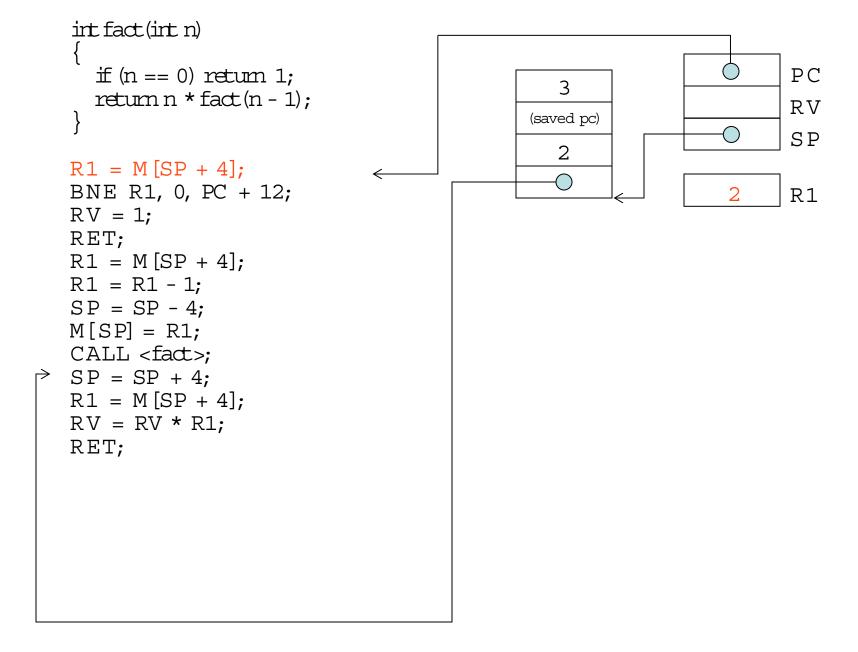
```
int fact (int n)
  if (n == 0) return 1;
  return n * fact(n - 1);
R1 = M[SP + 4];
BNE R1, 0, PC + 12;
RV = 1;
RET;
R1 = M[SP + 4];
R1 = R1 - 1;
SP = SP - 4;
M[SP] = R1;
CALL <fact>;
SP = SP + 4;
R1 = M[SP + 4];
RV = RV * R1;
RET;
```

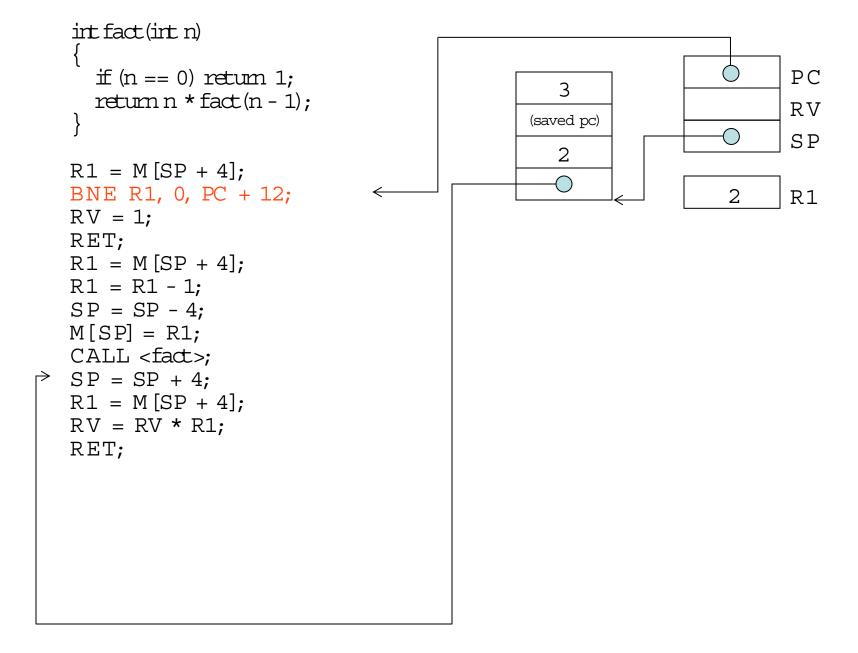


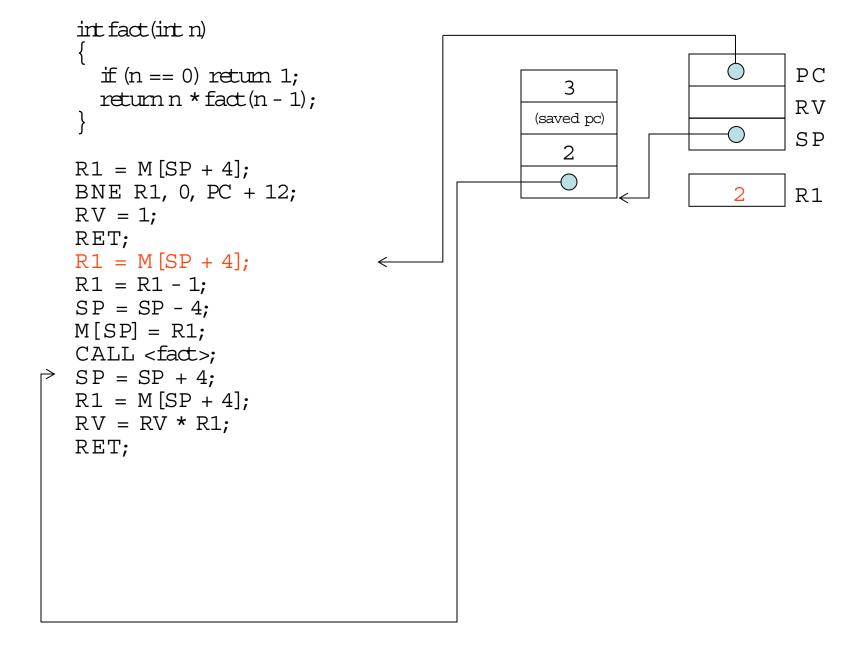
```
int fact (int n)
  if (n == 0) return 1;
  return n * fact(n - 1);
R1 = M[SP + 4];
BNE R1, 0, PC + 12;
RV = 1;
RET;
R1 = M[SP + 4];
R1 = R1 - 1;
SP = SP - 4;
M[SP] = R1;
CALL <fact>;
SP = SP + 4;
R1 = M[SP + 4];
RV = RV * R1;
RET;
```



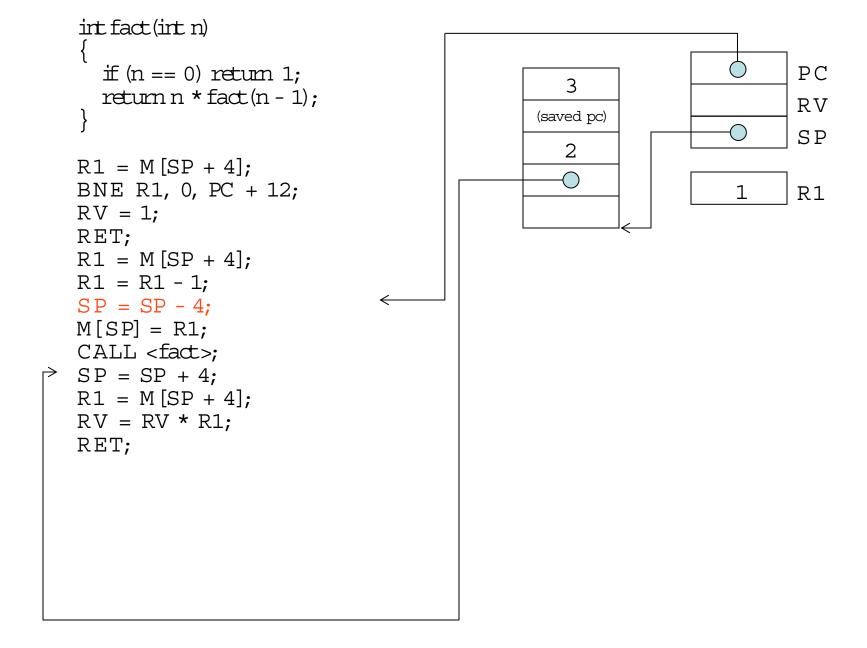
```
int fact (int n)
                                                                        PС
     if (n == 0) return 1;
                                                  3
     return n * fact(n - 1);
                                                                        RV
                                               (saved pc)
                                                                        SP
                                                  2
   R1 = M[SP + 4];
   BNE R1, 0, PC + 12;
                                                                  2
                                                                        R1
   RV = 1;
   RET;
   R1 = M[SP + 4];
   R1 = R1 - 1;
   SP = SP - 4;
   M[SP] = R1;
   CALL <fact>;
\Rightarrow SP = SP + 4;
   R1 = M[SP + 4];
   RV = RV * R1;
  RET;
```

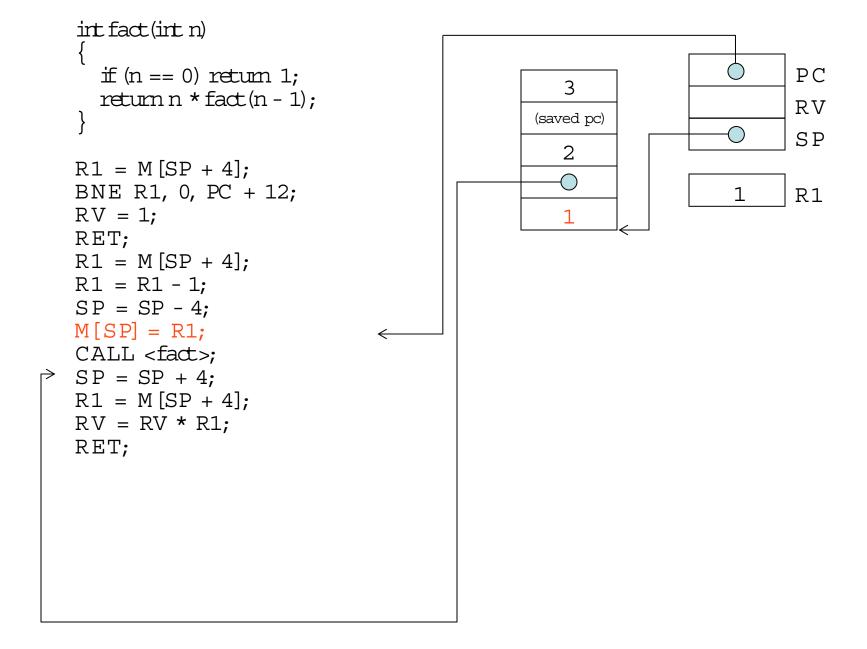


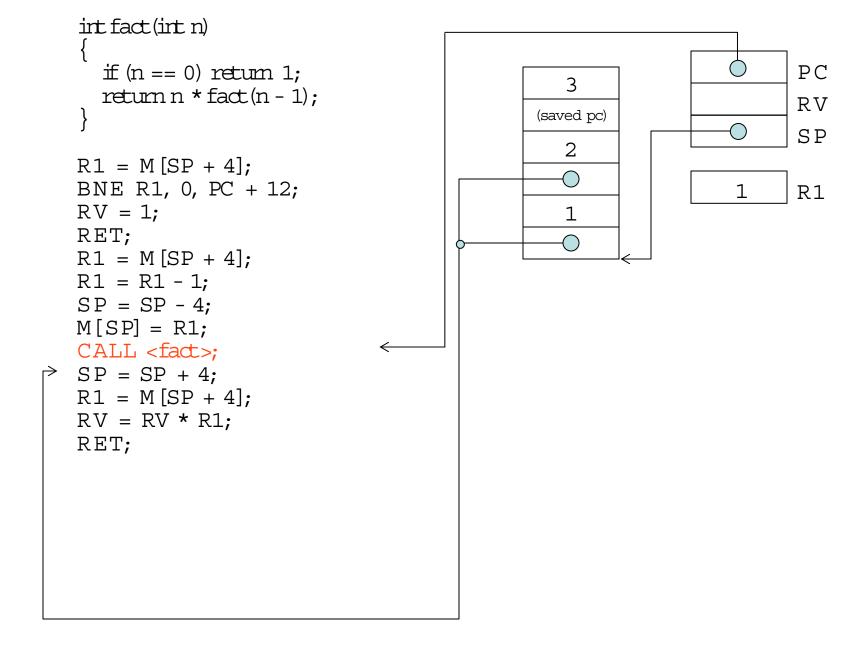


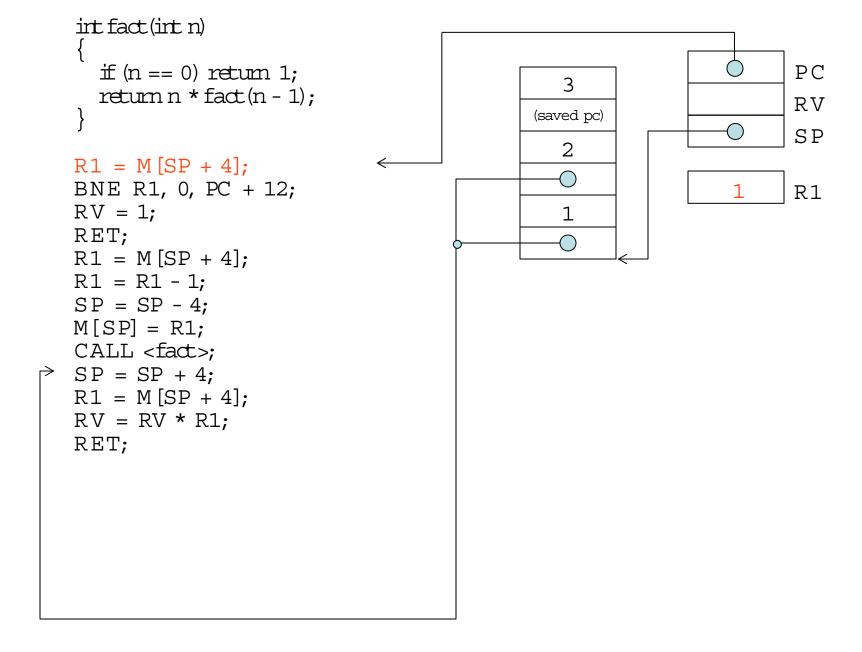


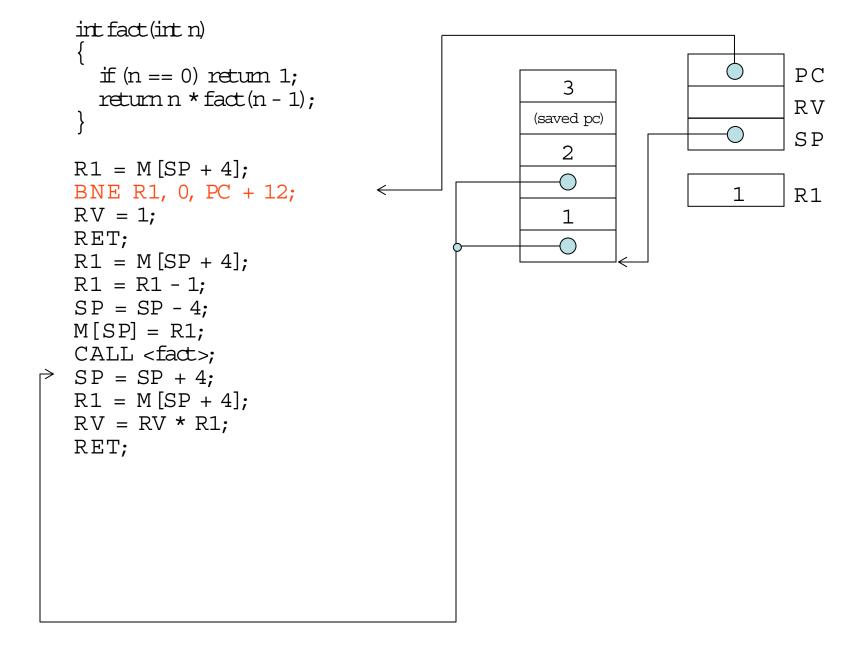
```
int fact (int n)
                                                                        PС
     if (n == 0) return 1;
                                                  3
     return n * fact(n - 1);
                                                                        RV
                                               (saved pc)
                                                                        SP
                                                  2
   R1 = M[SP + 4];
   BNE R1, 0, PC + 12;
                                                                        R1
   RV = 1;
   RET;
   R1 = M[SP + 4];
   R1 = R1 - 1;
   SP = SP - 4;
   M[SP] = R1;
   CALL <fact>;
\Rightarrow SP = SP + 4;
   R1 = M[SP + 4];
   RV = RV * R1;
  RET;
```

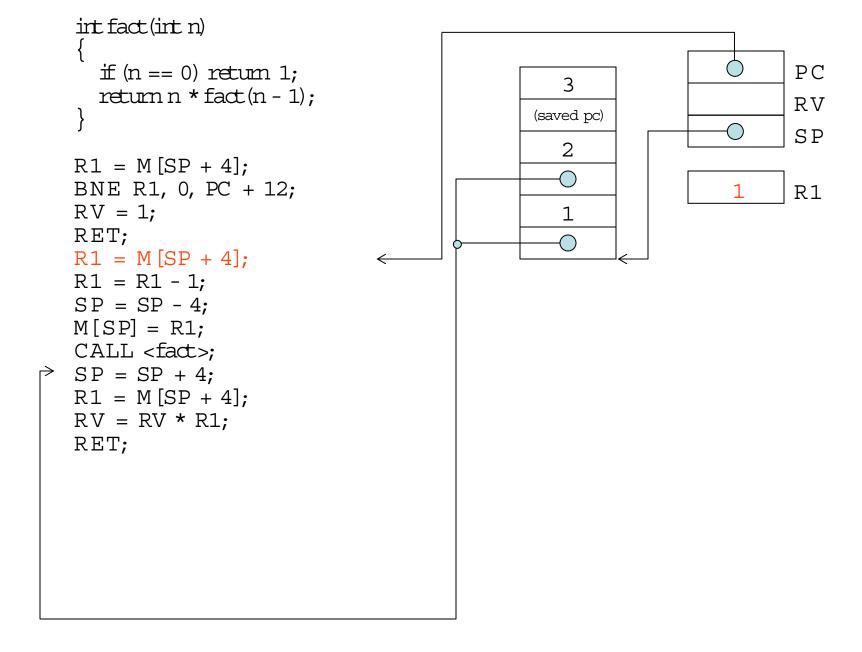


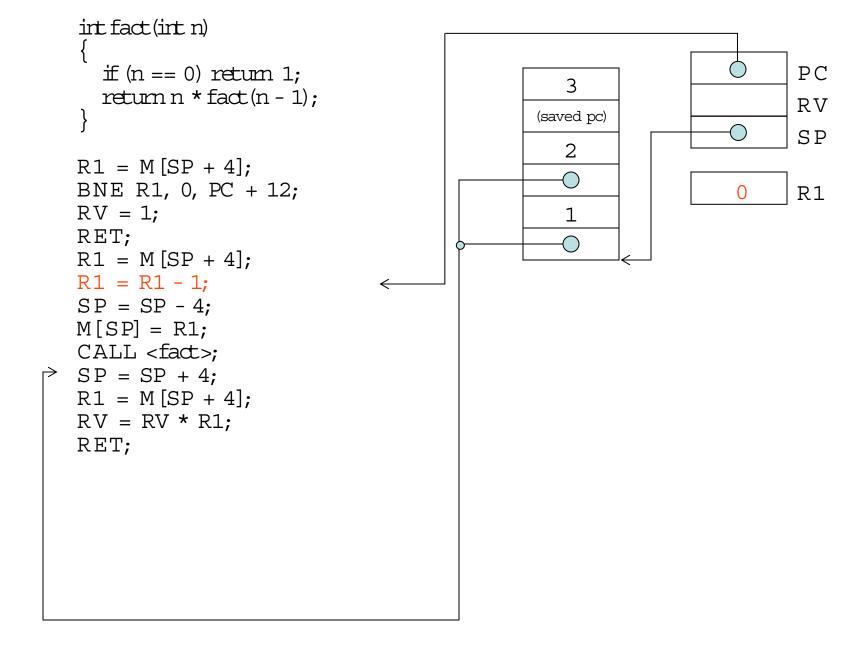


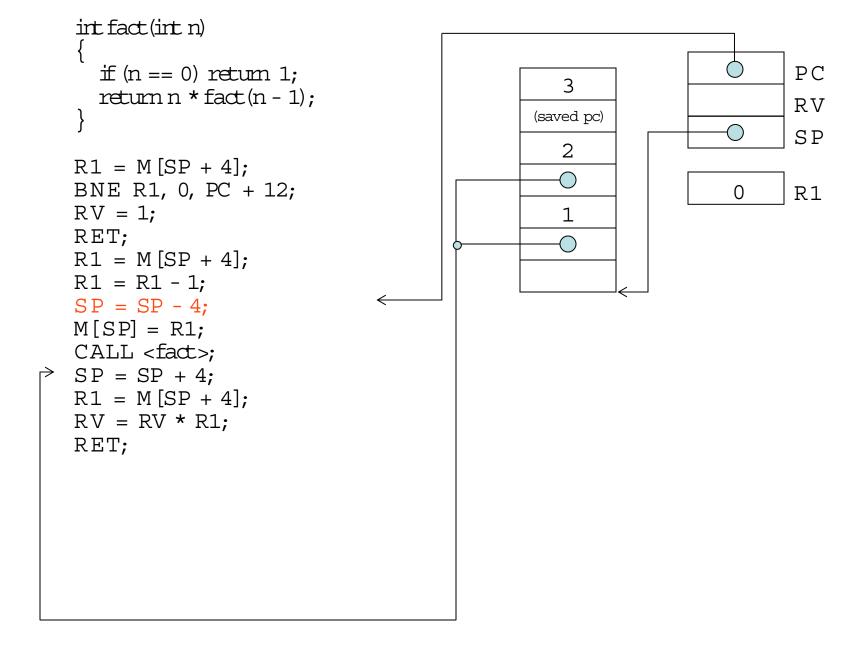


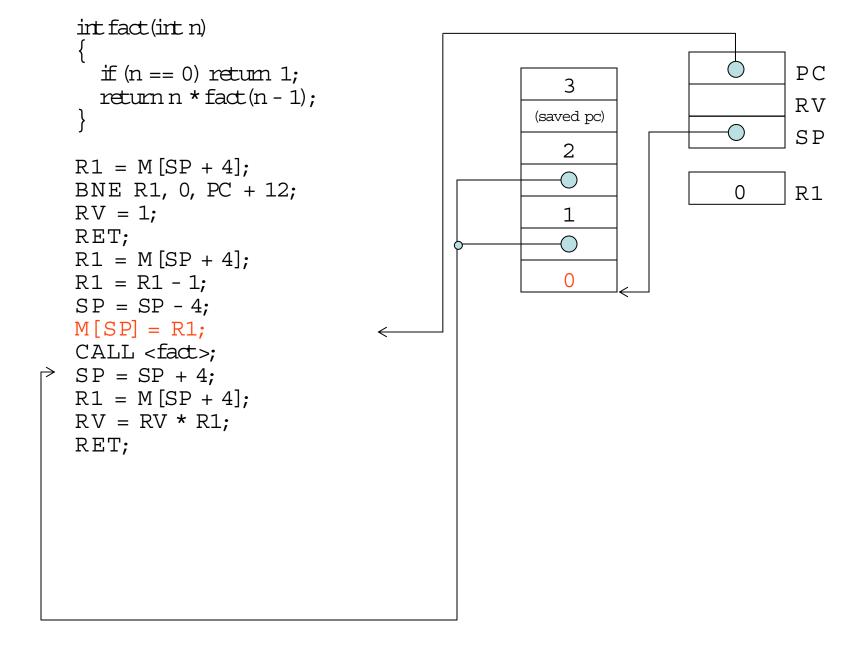


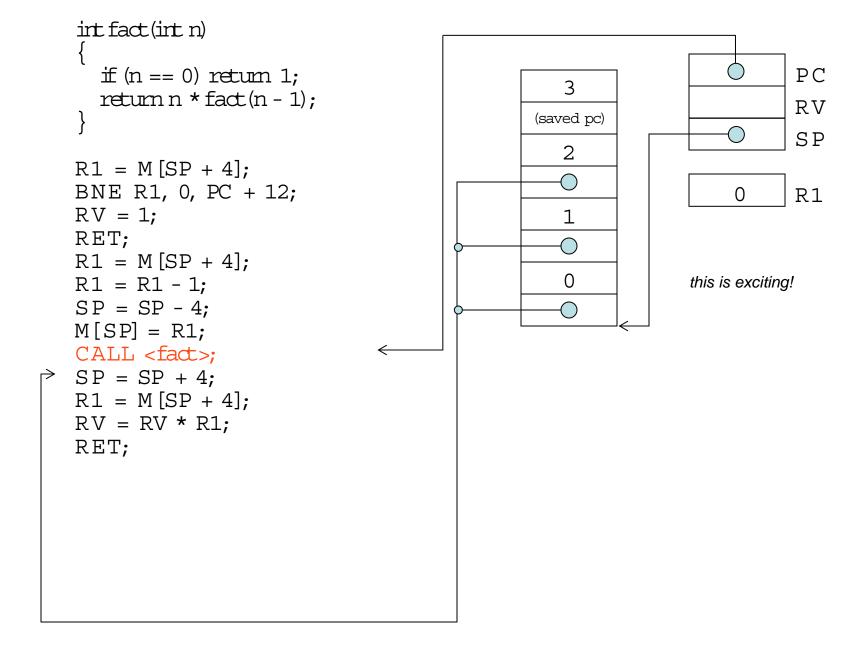


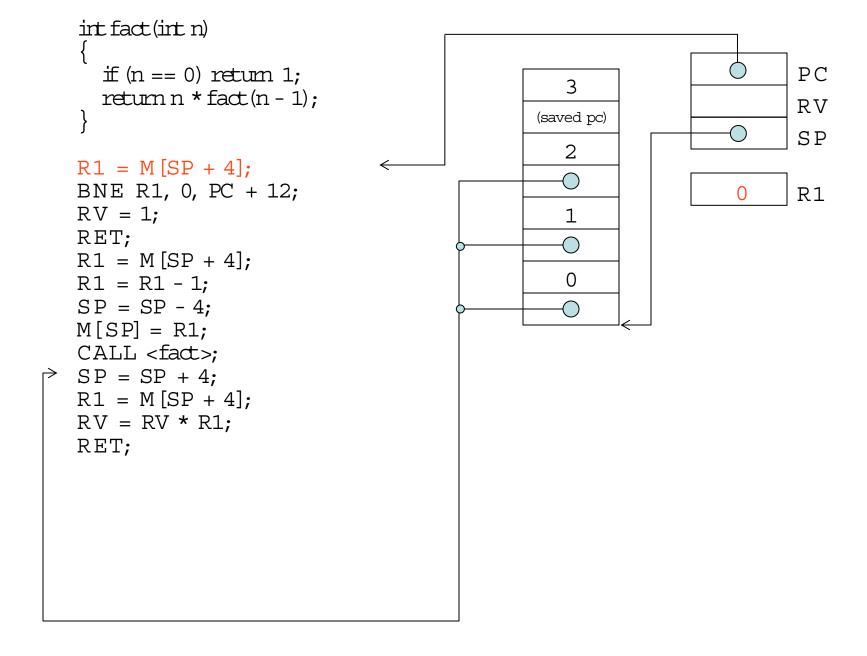


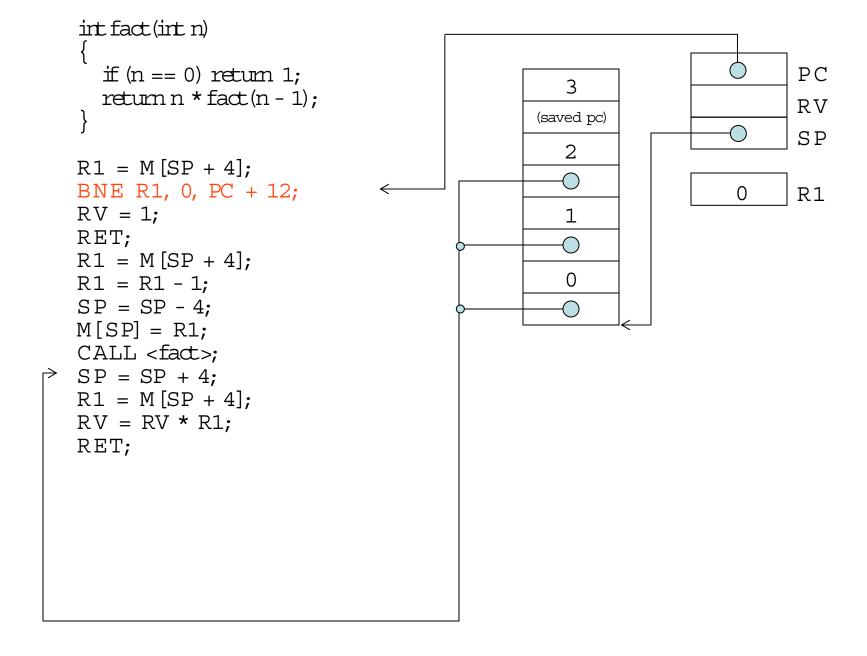


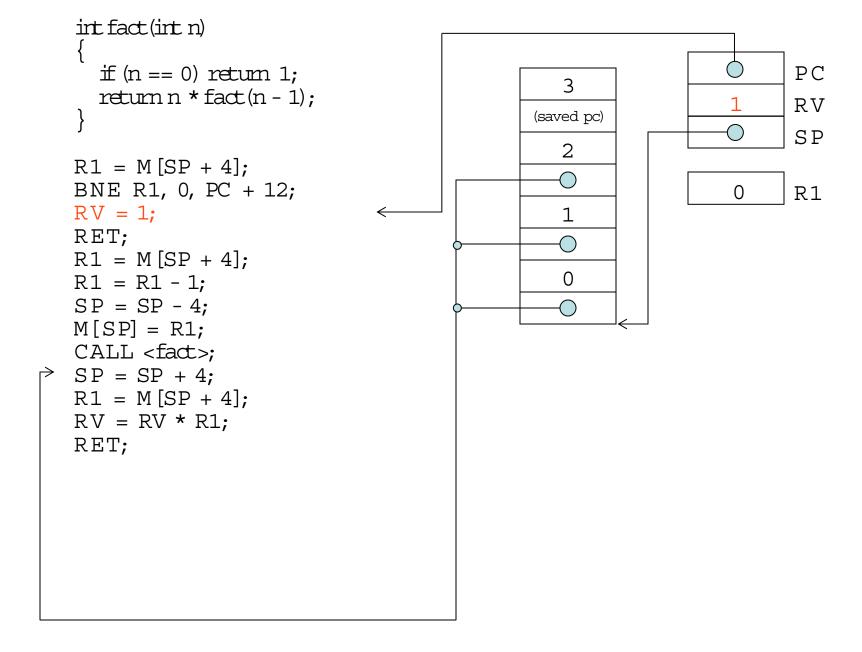


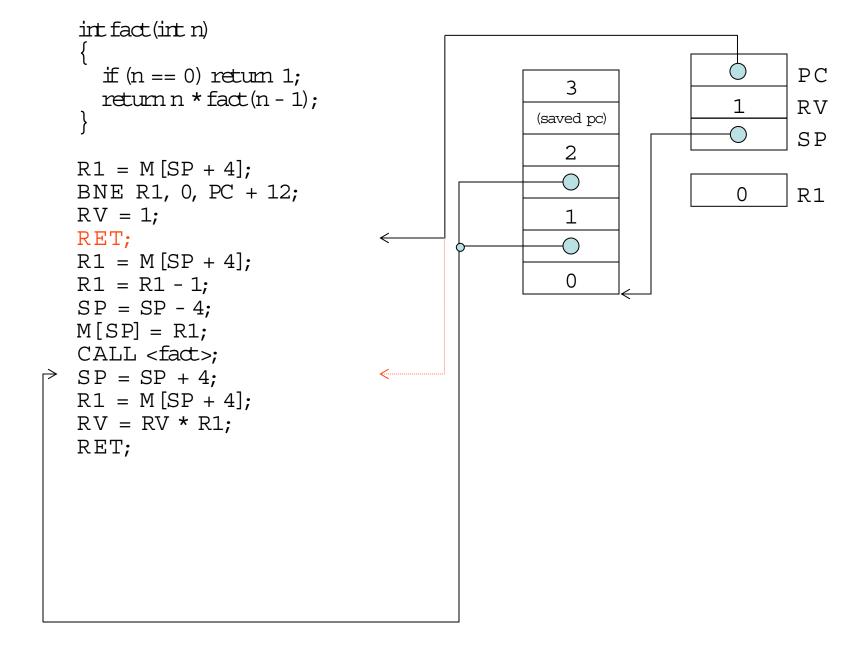


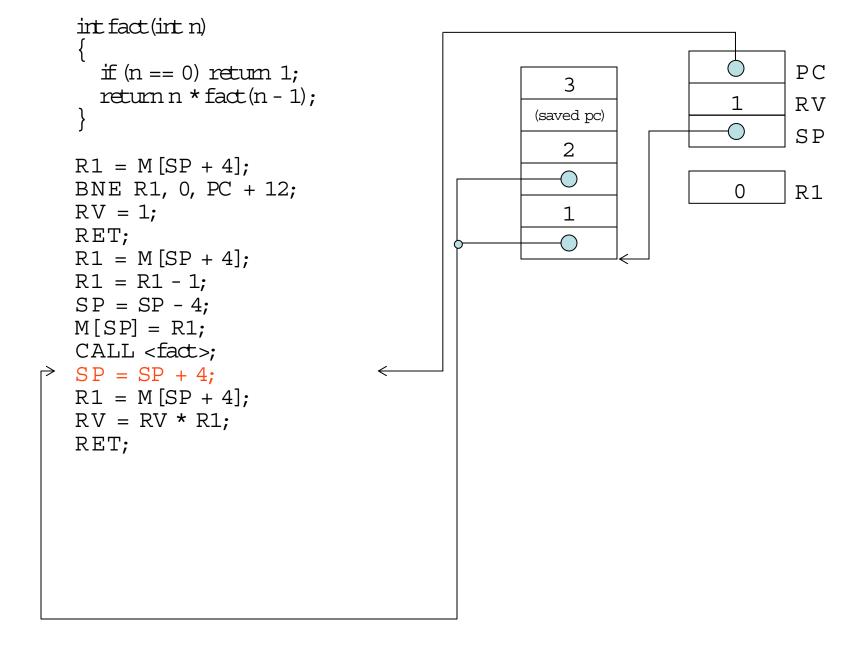


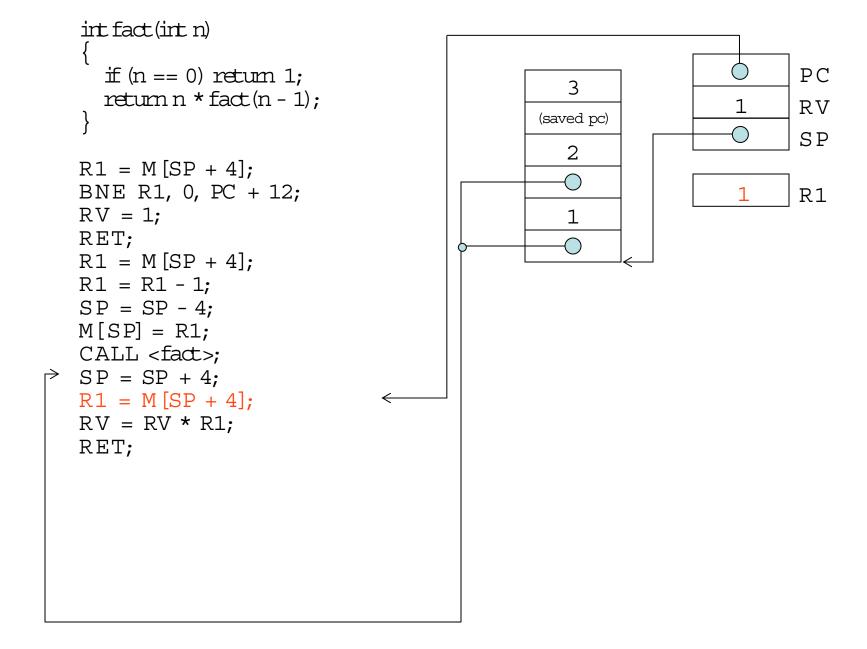


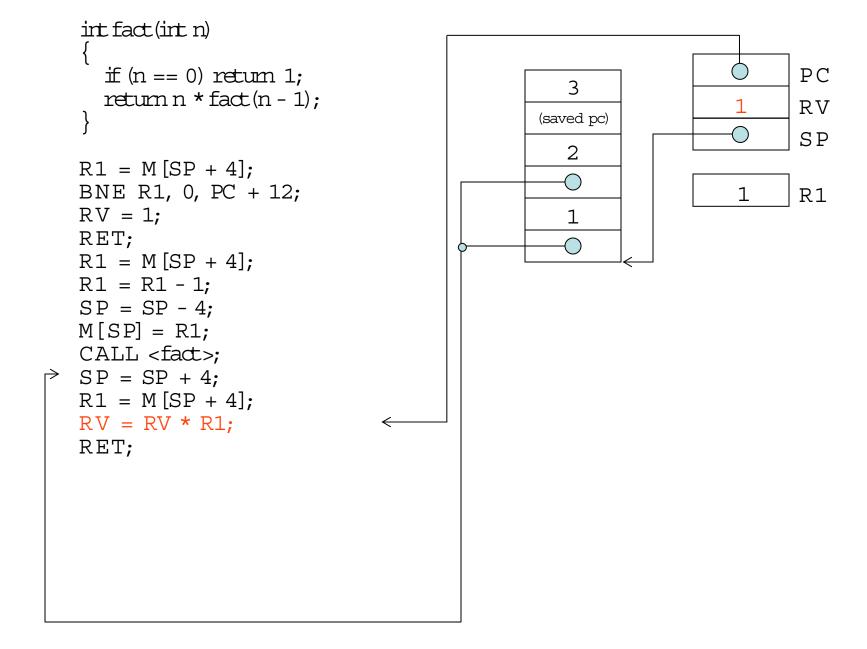


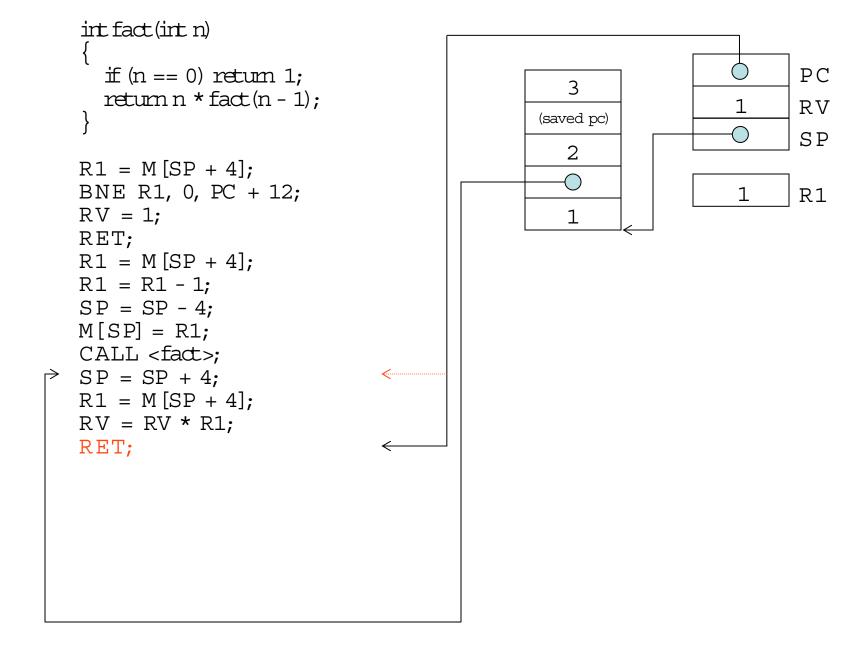












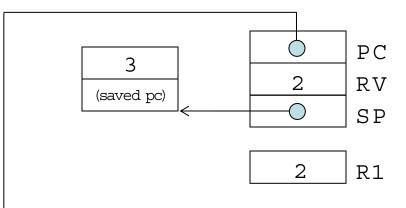
```
int fact (int n)
                                                                        PС
     if (n == 0) return 1;
                                                  3
     return n * fact(n - 1);
                                                                        RV
                                               (saved pc)
                                                                        SP
                                                  2
   R1 = M[SP + 4];
   BNE R1, 0, PC + 12;
                                                                  1
                                                                        R1
   RV = 1;
   RET;
   R1 = M[SP + 4];
   R1 = R1 - 1;
   SP = SP - 4;
   M[SP] = R1;
   CALL <fact>;
\Rightarrow SP = SP + 4;
   R1 = M[SP + 4];
   RV = RV * R1;
  RET;
```

```
int fact (int n)
                                                                        PС
     if (n == 0) return 1;
                                                  3
     return n * fact(n - 1);
                                                                        RV
                                               (saved pc)
                                                                        SP
                                                  2
   R1 = M[SP + 4];
   BNE R1, 0, PC + 12;
                                                                        R1
   RV = 1;
   RET;
   R1 = M[SP + 4];
   R1 = R1 - 1;
   SP = SP - 4;
   M[SP] = R1;
   CALL <fact>;
\Rightarrow SP = SP + 4;
   R1 = M[SP + 4];
   RV = RV * R1;
  RET;
```

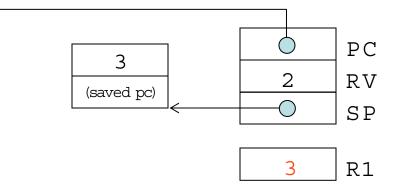
```
int fact (int n)
                                                                        PС
     if (n == 0) return 1;
                                                  3
     return n * fact(n - 1);
                                                                        RV
                                               (saved pc)
                                                                        SP
                                                  2
   R1 = M[SP + 4];
   BNE R1, 0, PC + 12;
                                                                  2
                                                                        R1
   RV = 1;
   RET;
   R1 = M[SP + 4];
   R1 = R1 - 1;
   SP = SP - 4;
   M[SP] = R1;
   CALL <fact>;
\Rightarrow SP = SP + 4;
   R1 = M[SP + 4];
   RV = RV * R1;
  RET;
```

```
int fact (int n)
                                                                   PС
  if (n == 0) return 1;
                                             3
  return n * fact(n - 1);
                                                                   RV
                                           (saved pc)
                                                                   SP
                                             2
R1 = M[SP + 4];
BNE R1, 0, PC + 12;
                                                             2
                                                                   R1
RV = 1;
RET;
R1 = M[SP + 4];
R1 = R1 - 1;
SP = SP - 4;
M[SP] = R1;
CALL <fact>;
SP = SP + 4;
R1 = M[SP + 4];
RV = RV * R1;
RET;
```

```
int fact (int n)
  if (n == 0) return 1;
  return n * fact(n - 1);
R1 = M[SP + 4];
BNE R1, 0, PC + 12;
RV = 1;
RET;
R1 = M[SP + 4];
R1 = R1 - 1;
SP = SP - 4;
M[SP] = R1;
CALL <fact>;
SP = SP + 4;
R1 = M[SP + 4];
RV = RV * R1;
RET;
```



```
int fact (int n)
  if (n == 0) return 1;
  return n * fact(n - 1);
R1 = M[SP + 4];
BNE R1, 0, PC + 12;
RV = 1;
RET;
R1 = M[SP + 4];
R1 = R1 - 1;
SP = SP - 4;
M[SP] = R1;
CALL <fact>;
SP = SP + 4;
R1 = M[SP + 4];
RV = RV * R1;
RET;
```



```
int fact (int n)
  if (n == 0) return 1;
  return n * fact(n - 1);
R1 = M[SP + 4];
BNE R1, 0, PC + 12;
RV = 1;
RET;
R1 = M[SP + 4];
R1 = R1 - 1;
SP = SP - 4;
M[SP] = R1;
CALL <fact>;
SP = SP + 4;
R1 = M[SP + 4];
RV = RV * R1;
RET;
```

