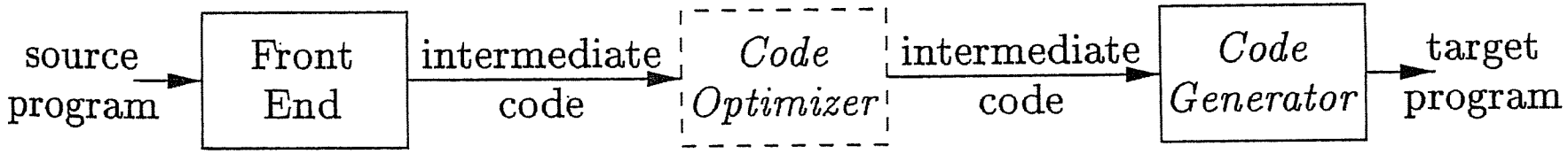


Figure 8.1: Position of code generator

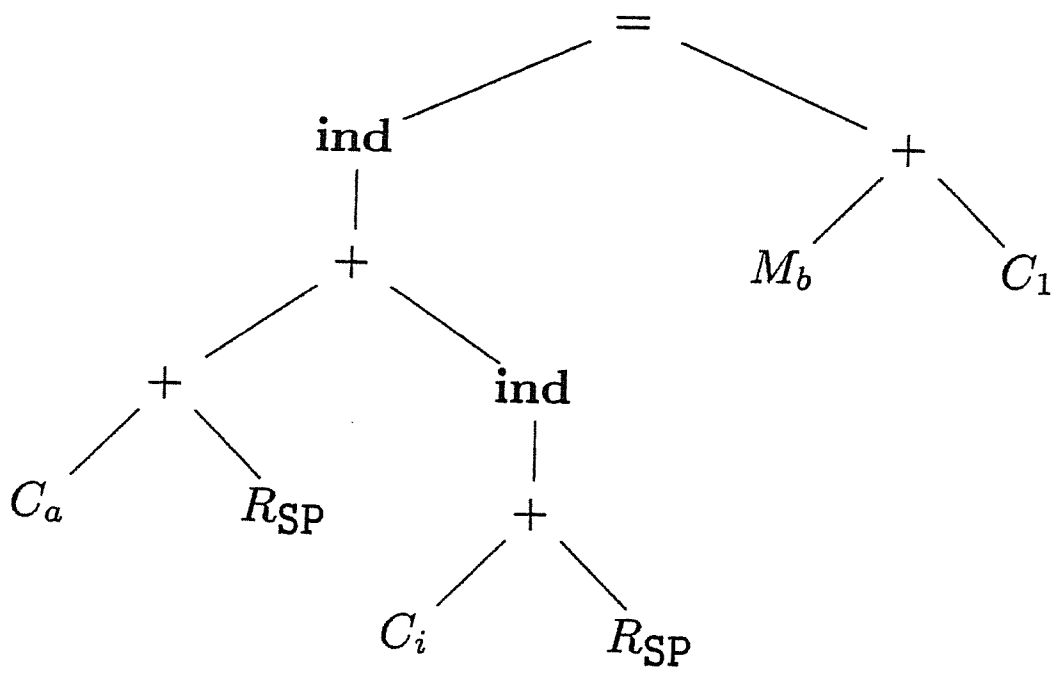


Figure 8.19: Intermediate-code tree for `a[i] = b + 1`

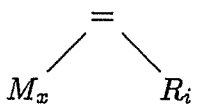
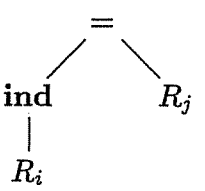
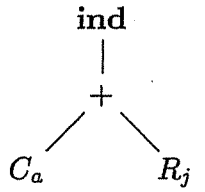
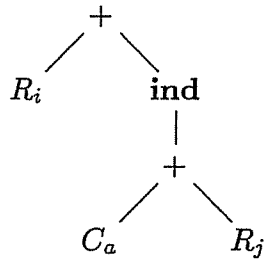
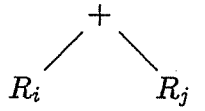
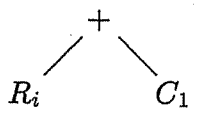
1)	$R_i \leftarrow C_a$	{ LD Ri, #a }
2)	$R_i \leftarrow M_x$	{ LD Ri, x }
3)	$M \leftarrow$ 	{ ST x, Ri }
4)	$M \leftarrow$ 	{ ST *Ri, Rj }
5)	$R_i \leftarrow$ 	{ LD Ri, a(Rj) }
6)	$R_i \leftarrow$ 	{ ADD Ri, Ri, a(Rj) }
7)	$R_i \leftarrow$ 	{ ADD Ri, Ri, Rj }
8)	$R_i \leftarrow$ 	{ INC Ri }

Figure 8.20: Tree-rewriting rules for some target-machine instructions

1)	$R_i \rightarrow \mathbf{c}_a$		{ LD Ri, #a }
2)	$R_i \rightarrow M_x$		{ LD Ri, x }
3)	$M \rightarrow = M_x\ R_i$		{ ST x, Ri }
4)	$M \rightarrow = \mathbf{ind}\ R_i\ R_j$		{ ST *Ri, Rj }
5)	$R_i \rightarrow \mathbf{ind} + \mathbf{c}_a\ R_j$		{ LD Ri, a(Rj) }
6)	$R_i \rightarrow + R_i\ \mathbf{ind} + \mathbf{c}_a\ R_j$		{ ADD Ri, Ri, a(Rj) }
7)	$R_i \rightarrow + R_i\ R_j$		{ ADD Ri, Ri, Rj }
8)	$R_i \rightarrow + R_i\ \mathbf{c}_1$		{ INC Ri }
9)	$R \rightarrow \mathbf{sp}$		
10)	$M \rightarrow \mathbf{m}$		

Figure 8.21: Syntax-directed translation scheme constructed from Fig. 8.20

```
 1)  i = 1
 2)  j = 1
 3)  t1 = 10 * i
 4)  t2 = t1 + j
 5)  t3 = 8 * t2
 6)  t4 = t3 - 88
 7)  a[t4] = 0.0
 8)  j = j + 1
 9)  if j <= 10 goto (3)
10)  i = i + 1
11)  if i <= 10 goto (2)
12)  i = 1
13)  t5 = i - 1
14)  t6 = 88 * t5
15)  a[t6] = 1.0
16)  i = i + 1
17)  if i <= 10 goto (13)
```

igure 8.7: Intermediate code to set a 10×10 matrix to an identity matrix

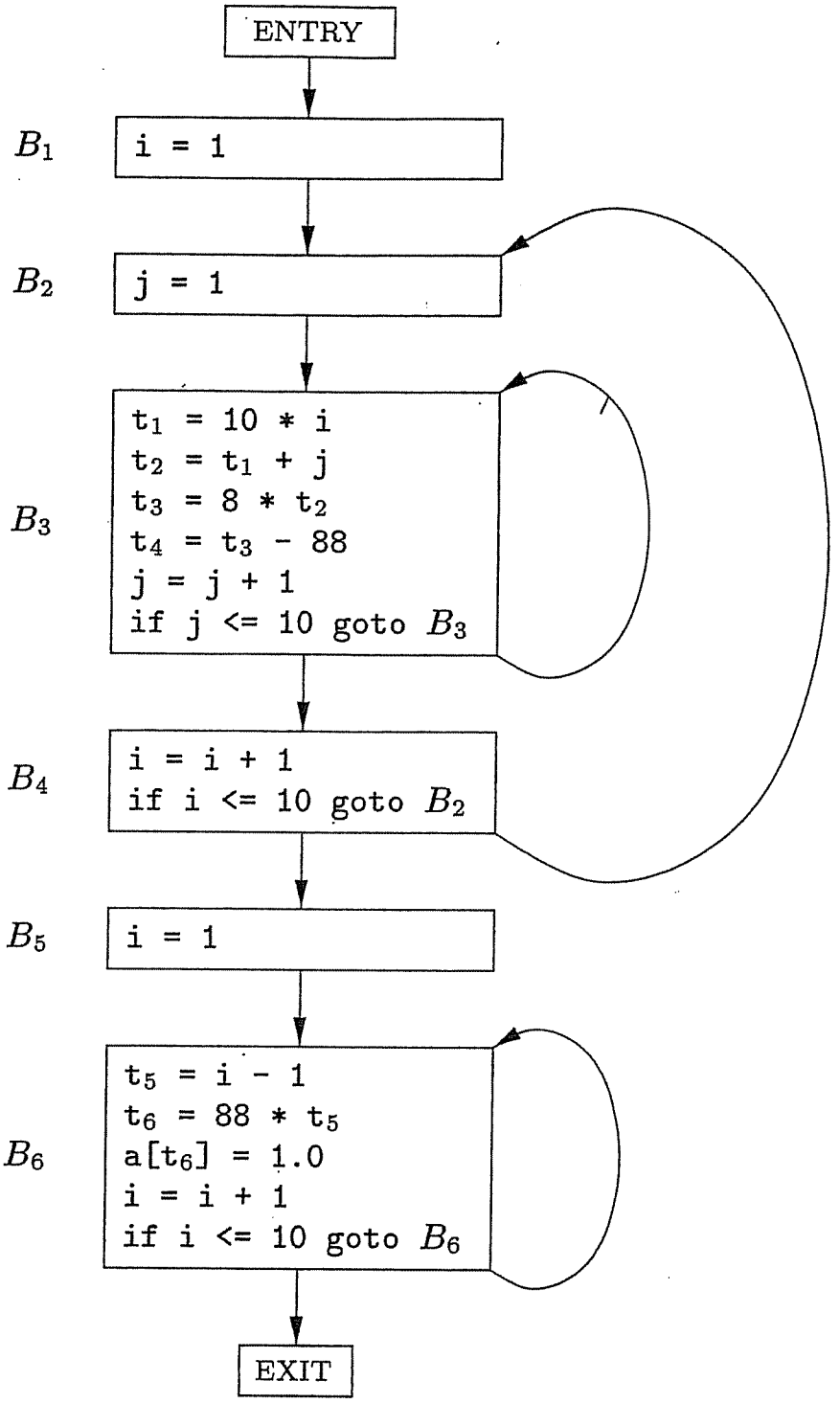


Figure 8.9: Flow graph from Fig. 8.7