

17. LABELS AND SWITCHES

17.1 go_to-statements

A program point (label) and a switch is uniquely defined by the following items:

Ordinary (label or switch):

Apparent block level and program address.

Virtual (label or switch):

Apparent block level and virtual index.

Actual parameter (label or switch):

Driver pointer and program address. (This is called a dynamic label).

Go to an ordinary label (except for local labels) and go to a formal label will be treated as equivalent since replacing the apparent block level BL by DDISPLAY (BL) for an ordinary label will give a case that may be handled by the formal go to procedure.

Thus only two routines in the runtime system will handle go to as a label:

GL (go to label)
GVL (go to virtual label)

The subroutine CONDDEL (which will determine whether a driver shall be deleted or not and perform the deletion) is used by GL.

For an actual parameter which is not an identifier, a thunk is created.

A designational expression is evaluated by TFL (take formal label).

For a switch, a switch calculation routine SWC is assumed to calculate a dynamic label (dp,pa) and enter the go to subroutine. This routine is not described here.

```
procedure condel (x); ref (driver) x;  
  begin  
    if x.md then  
      begin if not x.obj.PP.local classes then  
        begin if x.dot then deletenotice (x.drp);  
          deletenotice (x); x.obj.MDP :- none; end  
        else begin x.drex :- x.drp; x.pex :- none; x.acs :- none;  
          end  
        end  
      else if x.dot then begin deletenotice (x.drp);  
        deletenotice (x) end  
      else deletenotice (x);  
    end condel;
```

```
procedure GVL (bl,index); integer bl,index;  
  begin ref (program) k;  
    k :- DISPLAY (bl).PP.progaddr (index) qua program;  
    if k == none then error ("GVL",1);  
    GL (DISPLAY (bl),k)  
  end GVL;
```

```
procedure GL(b,m); ref (object) b; ref (program) m;  
  begin ref (driver) d; Boolean legal;  
    while CD.obj /= b or not CD.md do  
      begin if CD.rp then  
        begin d :- CD.drp;  
          if d == none then error ("GL",1);  
          legal := CD.pb;  
        end else d:- CD.drex;  
        condel (CD);  
        CD :- d;  
      end;  
    if not legal then error ("GL",2);  
    go to m;  
  end GL;
```