Status	Name	Length	Branches	Symbols
Flag	0	0	5	R14 R6 V1 V2 V3
Top	+10V	14	6	Q2 Q5 Q7 R2 R8 V1
Bot	-10V	12	5	Q8 R11 R13 R3 V2
Net	8	28	6	C2 C3 Q3 Q4 Q5 R9
NetR	A	18	6	Q6 Q7 Q8 R12 Ro V4
Net	10	24	5	C3 Q4 Q6 R10 R11
Net	4	14	4	C1 Q1 R2 R4
Net	11	2	3	Q5 Q7 R12
Net	12	2	3	Q6 Q8 R13
Net	2	4	3	Q2 R6 R7
Net	3	6	3	Q1 Q2 R3
Net	5	4	3	C1 R4 R5
Net	6	2	3	C2 Q3 R5
Net	9	6	3	Q4 R10 R9
Net	1	1	2	Q1 R1
Net	7	3	2	Q3 R8
Net	B	3	2	R7 V4
NetL	IN	1	2	R1 V3
---	---	---	---	---
Total:	18	136	33*	29

*NOTE: Branches are reported twice so Total = sum/2

Status Key	Connectivity	Optimization Weighting
Port:	port flags	zero length
Top:	top edge	vertical length only
Bot:	bottom edge	vertical length only
Net:	node to node	vertical and horizontal length
NetL:	left end node	pinned to left end
NetR:	right end node	pinned to right end

Schematic Symbol Field (H x V): 7 x 6

+	- -	- -	- -	- -	- -	- -	- -	+
:			R3			Q2		:
:		C1	C2	C3	Q1		Q3	:
:	Q4	Q5	Q6		Q7	Q8	R1	:
:	R2		R4		R5	R6	R7	:
:	R8	R9	R10	R11	R12	R13	Ro	:
:		V1	V2	V3	V4			:
+	- -	- -	- -	- -	- -	- -	- -	+

Iterations without improvement: 100 … Done!

```
ID    Xpos  Ypos  Len   Pins  NetNames
R1    1     1     4     2     Vcc 1
E1    2     2     6     4     in 0 1 0

Length
For i=1 to list.count do
 X
For i=1 to list.count do
  If NetNameMatch begin for self.netnames match
inc(Length, abs(x-X0)+abs(y-Y0))
```

```
procedure SortMemo(Memo: TMemo);
var
 SortList: TStringList;
begin
 {Create temporary string list}
 SortList := TStringList.Create;
 try
  {Put contents of memo into string list}
  SortList.Assign(Memo.Lines);
  {Sort the contents of the string list}
  SortList.Sort;
  {Place back in Memo}
  Memo.Lines.Assign(SortList);
 finally
  SortList.Free;
 end; {try..finally}
end;
```