The Lisp Interface.

1. The instructions generated by the compiler have the form

```
(start.calling (function symbol) (number of arguments))
(send.variable.Ai (Prolog temporary register number))...
(call.lisp (number of results expected))
(unify.local.Xn (temporary register number))...
```

2. The ultimate idea is that start.calling should prepare a Lisp stack frame, which is why the number of arguments appears there instead of being counted as the number of send.variable.Ai instructions or something.

3. Currently, the arguments are passed in the multiple value variables. The first few instructions go something like this:

```
(SETQQ QP.AV (           ; Argument Variables
    MV.RETURNER0
    ...
    MV.RETURNER15
))

start.to.call.lisp:
        (put.16 I (the argument number))
        (put.24 C (the function symbol))
        (put.24 R QP.AV)

send.variable.Ai:
        (SET (CAR (get.24 R)) (QP.lispify (get.Aval N*)))
        (put.24 R (CDR (get.24 R)))

call.lisp:
        (SETQ MV.RETURNER 0 (SELECTQ I
            ( 0 (APPLY* C))
            ...
            (15 (APPLY* MV.RETURNER0  MV.RETURNER1  MV.RETURNER2
                        ...
                        MV.RETURNER15))
            (SHOULDNT 'Too% many% arguments)
        ))
        (put.16 I (the number of results))
        (put.16 R QP.AV)
        (put.24 S (get.24 H))
        (increment.cell.pointer H I)
        (until (zero I)
            (QP.prologify (CAR (get.24 R)) (get.24 S))
        )
```

The rest is Prolog.

4. Key features:

A. There is a limit to the number of arguments which can be passed from Prolog to Lisp (16) and a similar limit to the number which can be returned from Lisp to Prolog. Neither of these limits means very much.

B. If any arguments or results are lists, then Lisp consing will be done when the arguments are passed and Prolog consing will be done when the results are returned.
C. However, no other consing is done. In particular, if we call get0/1, no consing will be done.
D. The objects which can be passed between Prolog and Lisp in either direction are symbols, numbers, and lists. Other Lisp constants will be supported later.