APPENDIX E. ACCELERATOR CONTROL PROGRAM

```
\ ========================================================
\ CPU #0C fact definitions
\ ========================================================
`hex 0C target

\ INPUT "RULES" ------------------------------------------
\ All of the analog inputs are software filtered.  These
\ rules are evaluated 18 times/second, by making them
\ dependent on the 18HZ fact.

HEsteer1x RULE:
    DOFORTH adc2  DOFORTH 2ndFilter
    DOFORTH ApplyScale  18 HENCE
;RULE
HEsteer1x 18HZ :dependent
HEsteer1x StdCoeffs
hex 82C decimal 300 HEsteer1x ScaleFactors

HEsteer1y RULE:
    DOFORTH adc3  DOFORTH 2ndFilter
    DOFORTH ApplyScale  18 HENCE
;RULE
HEsteer1y 18HZ :dependent
HEsteer1y StdCoeffs
hex 7E7 decimal 330 HEsteer1y ScaleFactors

HEsteer2x RULE:
    DOFORTH adc4  DOFORTH 2ndFilter
    DOFORTH ApplyScale  18 HENCE
;RULE
HEsteer2x 18HZ :dependent
HEsteer2x StdCoeffs
hex 753 decimal 435 HEsteer2x ScaleFactors

HEsteer2y RULE:
    DOFORTH adc5  DOFORTH 2ndFilter
    DOFORTH ApplyScale  18 HENCE
;RULE
HEsteer2y 18HZ :dependent
HEsteer2y StdCoeffs
hex 78C decimal 470 HEsteer2y ScaleFactors

CoronaPos RULE:
```

```
    DOFORTH adc6  DOFORTH 2ndFilter
    DOFORTH ApplyScale  18 HENCE
;RULE
CoronaPos 18HZ :dependent
CoronaPos StdCoeffs
hex   0 decimal 550 CoronaPos ScaleFactors

CoronaLoad RULE:
    DOFORTH adc7  DOFORTH 2ndFilter
    DOFORTH ApplyScale  18 HENCE
;RULE
CoronaLoad 18HZ :dependent
CoronaLoad StdCoeffs
hex  1D decimal 214 CoronaLoad ScaleFactors

NISvacuum RULE:
    DOFORTH adc8  DOFORTH 2ndFilter
    18 HENCE
;RULE
NISvacuum 18HZ :dependent
NISvacuum StdCoeffs

LEvacuum RULE:
    DOFORTH adc9  DOFORTH 2ndFilter
    18 HENCE
;RULE
LEvacuum 18HZ :dependent
LEvacuum StdCoeffs

HEvacuum RULE:
    DOFORTH adc10  DOFORTH 2ndFilter
    18 HENCE
;RULE
HEvacuum 18HZ :dependent
HEvacuum StdCoeffs

Ionpumps RULE:
    DOFORTH adc11  DOFORTH 2ndFilter
    18 HENCE
;RULE
Ionpumps 18HZ :dependent
Ionpumps StdCoeffs

\ OUTPUT RULES ------------------------------------------

\ OPERATING RULES ---------------------------------------
decimal
variable MaxExtension   220 MaxExtension !    ( 22" )
variable MinExtension    40 MinExtension !    (  4" )
variable DesiredCorona   30 DesiredCorona !   ( 30 uA)
```

```
variable ThreshCorona
variable OuterThr     5 OuterThr !  5 ThreshCorona !
variable InnerThr     2 InnerThr !  ( for hysteresis )
: +extend   -1 RLY6 0 RLY7  InnerThr @ ThreshCorona ! ;
: +retract  0 RLY6 -1 RLY7  InnerThr @ ThreshCorona ! ;
: 0points   0 RLY6  0 RLY7  OuterThr @ ThreshCorona ! ;

CoronaLoad-low RULE:
    INVOKE CoronaLoad
    DesiredCorona FETCH   ThreshCorona FETCH  -
    <
;RULE
CoronaLoad-low 1HZ :dependent
0 CoronaLoad-low !export

CoronaLoad-high RULE:
    INVOKE CoronaLoad
    DesiredCorona FETCH   ThreshCorona FETCH  +
    >
;RULE
CoronaLoad-high 1HZ :dependent
0 CoronaLoad-high !export

ExtendPoints RULE:
    CoronaLoad-low
    TermMV-high  AND
    INVOKE CoronaPos  MaxExtension FETCH <  AND
    CONCLUDES DOFORTH +extend
;RULE

RetractPoints RULE:
    CoronaLoad-high
    TermMV-low  AND
    INVOKE CoronaPos  MinExtension FETCH >  AND
    CONCLUDES DOFORTH +retract
;RULE

HoldPoints RULE:
    ExtendPoints
    RetractPoints OR  NOT
    CONCLUDES DOFORTH 0points
;RULE

\ Rules to turn off extend/retract motors incorporate
\ hysteresis by using a different test value.
\ Note also, < > for position test gives small hysteresis.
\
\ -ExtendPoints RULE:
\     INVOKE CoronaLoad   DesiredCorona FETCH   >
\     INVOKE CoronaPos  MaxExtension FETCH >  OR
```

```
\     DesiredMV 0=  OR
\     CONCLUDES DOFORTH -extend
\ ;RULE
\ -ExtendPoints 1HZ :dependent
\
\ -RetractPoints RULE:
\     INVOKE CoronaLoad   DesiredCorona FETCH   <
\     DesiredMV 0= NOT  AND
\     INVOKE CoronaPos  MinExtension FETCH <  OR
\     CONCLUDES DOFORTH -retract
\ ;RULE
\ -RetractPoints 1HZ :dependent

\ ============================================================
\ CPU #0D fact definitions
\ ============================================================
`hex 0D target

\ INPUT "RULES" ------------------------------------------
\ All of the analog inputs are software filtered.  These
\ rules are evaluated 18 times/second, by making them
\ dependent on the 18HZ fact.

HEchgV RULE:
    DOFORTH adc2  DOFORTH 2ndFilter
    DOFORTH ApplyScale  18 HENCE
;RULE
HEchgV 18HZ :dependent
HEchgV StdCoeffs
hex -0BD decimal 628 HEchgV ScaleFactors

HEchgA RULE:
    DOFORTH adc3  DOFORTH 2ndFilter
    DOFORTH ApplyScale  18 HENCE
;RULE
HEchgA 18HZ :dependent
HEchgA StdCoeffs
hex   4 decimal 250 HEchgA ScaleFactors

LEchgV RULE:
    DOFORTH adc4  DOFORTH 2ndFilter
    DOFORTH ApplyScale  18 HENCE
;RULE
LEchgV 18HZ :dependent
LEchgV StdCoeffs
hex  5D decimal 700 LEchgV ScaleFactors

LEchgA RULE:
    DOFORTH adc5  DOFORTH 2ndFilter
    DOFORTH ApplyScale  18 HENCE
```

```
;RULE
LEchgA 18HZ :dependent
LEchgA StdCoeffs
hex -0B3 decimal 250 LEchgA ScaleFactors

HEcolA RULE:
    DOFORTH adc6  DOFORTH 2ndFilter
    DOFORTH ApplyScale  18 HENCE
;RULE
HEcolA 18HZ :dependent
HEcolA StdCoeffs
hex  72 decimal 190 HEcolA ScaleFactors

LEcolA RULE:
    DOFORTH adc7  DOFORTH 2ndFilter
    DOFORTH ApplyScale  18 HENCE
;RULE
LEcolA 18HZ :dependent
LEcolA StdCoeffs
hex 0D5 decimal 270 LEcolA ScaleFactors

TermMV RULE:
    DOFORTH adc8  DOFORTH 2ndFilter
    DOFORTH ApplyScale  18 HENCE
;RULE
TermMV 18HZ :dependent
TermMV StdCoeffs
hex  12 decimal 2100 TermMV ScaleFactors

HEquad1 RULE:
    DOFORTH adc10  DOFORTH 2ndFilter
    18 HENCE
;RULE
HEquad1 18HZ :dependent
HEquad1 StdCoeffs

HEquad2 RULE:
    DOFORTH adc11  DOFORTH 2ndFilter
    18 HENCE
;RULE
HEquad2 18HZ :dependent
HEquad2 StdCoeffs

\ OUTPUT RULES ------------------------------------------------
\ The analog outputs are set via closed-loop PID control.
\ These rules are evaluated 18 times/second, by making them
\ dependent on the 18HZ fact.

HEchgVOut RULE:
    HEchgV  DOFORTH DROP     \ sensor value
```

```
    DOFORTH ApplyPID  DOFORTH dac1
    DOFORTH OLDFACT
;RULE
hex 100 SET HEchgVOut .Igain
   1000 SET HEchgVOut .Pgain
      0 SET HEchgVOut .Dgain
      0 SET HEchgVOut .sumerror
      0 SET HEchgVOut .error

LEchgVOut RULE:
    LEchgV  DOFORTH DROP     \ sensor value
    DOFORTH ApplyPID  DOFORTH dac2
    DOFORTH OLDFACT
;RULE
hex 100 SET LEchgVOut .Igain
   1000 SET LEchgVOut .Pgain
      0 SET LEchgVOut .Dgain
      0 SET LEchgVOut .sumerror
      0 SET LEchgVOut .error

: resetpid  0 HEchgVOut objadr .sumerror !
            0 LEchgVOut objadr .sumerror ! ;

\ OPERATING RULES ----------------------------------------
\ terminal voltage control
\ these rules are evaluated every 1 second
decimal
VARIABLE Setpoint*10    300 Setpoint*10 !
VARIABLE Increment       5 Increment !
VARIABLE Tweak           1 Tweak !      ( fine adjustment )
VARIABLE Threshold      20 Threshold !  ( 0.20 MV )
VARIABLE TweakThresh     1 TweakThresh !  ( 0.01 MV )
VARIABLE RateTrip       10 RateTrip !   ( 0.10 MV/sec max )
VARIABLE MVTrip        500 MVTrip !     ( 5.00 MV max )
VARIABLE SparkTrip     -50 SparkTrip !  ( 0.50 MV drop )

: le  18 hence lechgvout :assert ;
: he  18 hence hechgvout :assert ;
: mv  forever desiredmv :assert ;   0 DesiredMV !export

\ emergency shutdown
: Panic
    0 mv  0 le  0 he    \ all setpoints to zero  ***TEMP***
    ;

: -chg  ( increment -- )
    Setpoint*10 @       ( current value)
    SWAP -              ( decrease, )
    300 MAX             (  but not less than 30)
    DUP Setpoint*10 !
```

```
    DUP 10 / 18 HENCE  LEchgVOut :assert
    5 + 10 / 18 HENCE  HEchgVout :assert ;   ( staggered )

: --chargeKV  Increment @ -chg ;
: -chargeKV   Tweak @ -chg ;

: +chg  ( increment -- )
    Setpoint*10 @       ( current value)
    SWAP +              ( increase, )
    1500 MIN            (  but not less than 150)
    DUP Setpoint*10 !
    DUP 10 / 18 HENCE  LEchgVOut :assert
    5 + 10 / 18 HENCE  HEchgVout :assert ;

: ++chargeKV  Increment @ +chg ;
: +chargeKV   Tweak @ +chg ;

VARIABLE PrevMV             \ previous TermMV reading
: DeltaMVEval
    TermMV :invoke swap          \ current TermMV reading
    DUP PrevMV @ -
    SWAP PrevMV !
    swap ;                  ( -- value time )


DeltaMV RULE:              \ change in Terminal MV
    DOFORTH DeltaMVEval
;RULE
DeltaMV 1HZ :dependent

TermMV-low RULE:         \ true if Desired>Term by at least .20 MV
    DesiredMV  INVOKE TermMV  -
    Threshold FETCH  >
;RULE
TermMV-low 1HZ :dependent
0 TermMV-low !export

TermMV-high RULE:        \ true if Desired<Term by at least .20 MV
    INVOKE TermMV  DesiredMV -
    Threshold FETCH  >
;RULE
TermMV-high 1HZ :dependent
0 TermMV-high !export

TermMV-bitlow RULE:      \ true if Desired>Term by at least .01 MV
    DesiredMV  INVOKE TermMV  -
    TweakThresh FETCH  >
;RULE
TermMV-bitlow 1HZ :dependent
```

```
TermMV-bithigh RULE:     \ true if Desired<Term by at least .01 MV
    INVOKE TermMV  DesiredMV -
    TweakThresh FETCH  >
;RULE
TermMV-bithigh 1HZ :dependent

TermMV-trip RULE:
    INVOKE TermMV  MVTrip FETCH  >
    CONCLUDES DOFORTH Panic
;RULE
TermMV-trip 1HZ :dependent

Spark RULE:
    DeltaMV  SparkTrip FETCH  <
    CONCLUDES DOFORTH Panic
;RULE

ChargeTooFast RULE:
    TermMV-low
    INVOKE DeltaMV  RateTrip FETCH  >  AND
    CONCLUDES DOFORTH --chargeKV
;RULE

MoreTweak RULE:
    INVOKE TermMV-bitlow
    TermMV-low NOT  AND
    CONCLUDES DOFORTH +chargekv
;RULE

LessTweak RULE:
    INVOKE TermMV-bithigh
    TermMV-high NOT  AND
    CONCLUDES DOFORTH -chargekv
;RULE

MoreCharge RULE:
    CoronaLoad-high NOT
    TermMV-low   AND
    CONCLUDES DOFORTH ++chargeKV
;RULE

LessCharge RULE:
    CoronaLoad-low NOT
    TermMV-high   AND
    CONCLUDES DOFORTH --chargeKV
;RULE
```