```
huns = 0
tens = 0
ones = 0
SEROUT 1, 16780, [254, 1]
SEROUT 1, 16780, ["ENTER FILL LOOP"]
PAUSE 2000

GOSUB fill_loop

SEROUT 1, 16780, [254, 1]           'purge
'SEROUT 1, 16780, ["CLOSE SUPPLY"]
SEROUT 1, 16780, [254, 192]
SEROUT 1, 16780, ["IN PURGE NOW"]
LOW 5
LOW 6
PAUSE 5000
HIGH 5
HIGH 6
GOTO begin

nodata:
PAUSE 5000
HIGH 5
HIGH 6
GOTO begin

z1digit:
SEROUT 1, 16780, [254,140]
SEROUT 1, 16780, ["z1df"]
PAUSE 100
act_gas = (huns * 100)
act_gas = act_gas + (tens * 10)
act_gas = act_gas + ones
GW = act_gas
act_gas = act_gas - tare
GOTO ret_fr_z1d

digit:
SEROUT 1, 16780, [254,140]
SEROUT 1, 16780, ["dig"]
PAUSE 100
act_gas = (huns * 100)
act_gas = act_gas + (tens * 10)
act_gas = act_gas + ones
tare = act_gas
GOTO ret_digit

fill_loop:
received:
  z = 1
  FOR z = 1 TO 255
    SEROUT 1, 16780, [254,140] 'show iteration
    SEROUT 1, 16780, [DEC z]
    act_gas = 0
    ones = 0
    tens = 0
    huns = 0
    thous = 0
    SEROUT 2, 16780, ["P"]    'send command to fire output
    SERIN 0, 16780, [junk, junk, junk, junk, c, f, junk, h, i, junk, junk, junk] 'Read the running digits into variables. g should always be a decimal point
    LOOKDOWN i, [48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58], ones   'put CO2 ones ascii chars into single bytes representing  decimal digits
    LOOKDOWN h, [48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58], tens  'put CO2 tens ascii chars into single bytes representing decimal digits
    LOOKDOWN f, [48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58], huns  'put CO2 hundreds ascii chars into single bytes representing decimal digits
    LOOKDOWN c, [48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58], thous ' thousands

      IF thous = 0 THEN z1digit
        act_gas =  (thous * 1000)          'build running net fill
        act_gas = act_gas + (huns * 100)
        act_gas = act_gas + (tens * 10)
        act_gas = act_gas + ones
        IF z = 1 AND text_in1 = 0 THEN tareout
    newtare:
      GW = act_gas                  'act_gas now = scale reading value
      act_gas = act_gas - tare      'subtract  cyl & hose weight in tare var. *-*-*-* act_gas now = gas only *-*-*-*

                        'GW now = act_gas ( actual fill wt for this cycle)
    ret_fr_z1d:
      SEROUT 1, 16780, [254,140] 'show routine
      SEROUT 1, 16780, ["z6"]
      'SEROUT 1, 16780, [254,1]
      SEROUT 1, 16780, ["gas_sp=", DEC gas_sp]
      SEROUT 1, 16780, [254,192]
      SEROUT 1, 16780, ["act_gas=", DEC act_gas]
      PAUSE 1000
      IF text_in1 = 0 THEN out_n2     'IF using an N2 set point
       SEROUT 16, 16780, ["1", "254", act_gas.HIGHBYTE, act_gas.LOWBYTE]' return actual fill value to vb form via serial  - progressive display

    ret_out_n2:
      SEROUT 1, 16780, [254,140] 'show routine
      SEROUT 1, 16780, ["out"]
      IF act_gas >= gas_sp THEN full
      error = gas_sp - act_gas      'calculate simple error (deviation from set point) value
      IF error <0 THEN zerror
      cterror:
      SEROUT 1, 16780, [254,140] 'show loop
      SEROUT 1, 16780, ["zz"]
      IF error = 0 THEN full
      pulse_len = error * 25
        IF pulse_len > 2000 THEN limit_pulse      'variable used as a constant - this determines the length of valve pulses

      shortened_pulse:              'need to differentiate which valve to control
        SEROUT 1, 16780, [254,140] 'show loop
        SEROUT 1, 16780, ["up"]
        IF error <= 0 THEN full     ' proportional cycle calculated here
        IF text_in1 = 0 THEN N2valve   'check flag, decide which valve to output to
        LOW 5                       'if flag not cleared, fire co2 valve on pin 5
        PAUSE pulse_len
        HIGH 5
        PAUSE proportion            'proportion is CON

      ret_valves:
        SEROUT 1, 16780, [254,140] 'show routine
        SEROUT 1, 16780, ["rv"]

      NEXT                          'loop return for 255 pass weigh/fill/SEROUT cycle
      z = 0

    full:                           'jump out of fill loop to here when full. Add conditional display text here for "could not completely fill cyl"
      SEROUT 1, 16780, [254, 1]
      SEROUT 1, 16780, ["gas_sp=", DEC gas_sp]
      SEROUT 1, 16780, [254,140] 'show loop
      SEROUT 1, 16780, ["full"]
      SEROUT 1, 16780, [254, 192]
      SEROUT 1, 16780, [DEC act_gas]
      HIGH 5                        'close co2 valve
      HIGH 6                        'close n2 valve

    'No need to differentiate whether CO2 or N2 - this sends address to change display text only.
    'IF text_in1 = 0 THEN out_n2

      SEROUT 16, 16780, ["1", "250", act_gas.HIGHBYTE, act_gas.LOWBYTE]' used to send ascii 48 -- case  displays instruction to open purge valve
      PAUSE 3000

      'SEROUT 1, 16780, [254, 1]
        SEROUT 1, 16780, ["   GAS FILLED"]
        'PAUSE 1000                  'display text dwell time
        SEROUT 1, 16780, [254, 1]    'clear display
        SEROUT 1, 16780, ["LP PURGE HOSE"]
        PAUSE 6000

    IF text_in1 = 0 THEN begin

    ********************************** END OF GAS FILL LOOP ***********************************************
    ********************************** SET UP FOR N2 FILL HERE ********************************
    error = 0
    'gas_sp = 0
    ones = 0
    tens = 0
    huns = 0
    thous = 0
    act_gas = 0
    text_in1 = 0                    'recycled Byte variable  - use as flag, when cleared, CO2 cycle is complete, proceed to fill N2
    text_in2 = 2                    'recycle this variable for use as N2 offset (wt of n2 in hose)
    SERIN 16, 1780, [WAIT(255), junk, junk, junk, junk, junk]
    SEROUT 1, 16780, [254,1]
    SEROUT 1, 16780, ["   FILLING N2"]
    PAUSE 2000
    'build N2 set point - always 2 or less places
    LOOKDOWN text_in4, [48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58], tens
    LOOKDOWN text_in5, [48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58], ones
    'text_in4 & text_in 5 may now be recycled
    gas_sp = (tens * 10)
    gas_sp = gas_sp + ones  'N2 set point will always be 1 or 2 places
    gas_sp = gas_sp + text_in2    'remember -- text_in2 has been recycled - is not a sp now!

    SEROUT 1, 16780, [254,1]
    SEROUT 1, 16780, ["N_sp=", DEC gas_sp]
    PAUSE 2000
    tens = 0
    ones = 0
    badN2:
    RETURN
```