```
' uMITE MMBASIC OPERATED MICROPROCESSOR PHASE CONTROLLED BATTERY CHARGER
' The battery charger's output voltage (& current limit) is controlled by a MicroMite (uMite)
microcontroller
' which controls the firing angle of one or 2 thyristors (SCR's) in the output circuit. The later the
' firing occurs in the AC cycle, the higher the O/P volts / current.
' A short trigger pulse is generated around the zero point of a rectified AC waveform. This trigger pulse is
' fed to two inputs on the uMite configured as counters. One input triggers the "OneShot" pulse generator
which
' is used to eventually trigger the SCR's. The other is used to count the trigers & on every 10th one
(100mSec), voltage &
' current readings are taken, values calculated & displayed & OneShot set. MMBASIC is not quick enough to
sample faster
' The uMite also generates high frequency (25kHz) PWM pulses which are gated by the OneShot pulse, fed fed
to a driver
' circuit & then to the SCR('s).
' In addition, the uMite generates a "Heatbeat" pulse which is fed to a detector circut. If the program
stops, the trigger
' pulses are inhibited. (the PWM & Oneshot pulse will continue to be produced even if the UMite's program
crashes or stops.
' A sample of the output voltage is scaled down & filtered & fed to an ADC input of the uMite & the value
is measured & scaled.
' Similarly, an output from the "Voltage Set" control is fed to another ADC input, measured & scaled. The
difference between the
' two voltages is used to determine the amount of delay adjustment required to bring the two values
together.
' The output current is measured by a "High Side" circuit which produces a voltage proportional to the
output current. This voltage
' is also measured by an ADC input & compared to another input set by the "Current Set" control. The
difference between the
' two is used to tell if the O/P current is greater than the set value. If it is, the firing angle is
adjusted until equilibrium
' is achieved. NOTE, the O/P current is checked before the O/P voltage & has priority.

' Sept 3rd, 2018. 2 SCR Hardwarwe Drive. SCR's replaced power diodes.
' more earth connections added to boards
' Sept, 12 PWM_Phase_12V_20A_02.bas Experimenting with synchronising measurementd in I/P trig
'   Counting input pulses in PIN(16) & measuring every 10 worked well. O/P very stable.
' PWM_Phase_12V_20A_03.bas  Prog adjust for 20A.
' PWM_Phase_12V_20A_04.bas  Limit$ changed to Mode$. NOT YET UPLOADED


OPTION AUTORUN ON               'program runs on startup
LCD INIT 23, 22, 21, 18, 14, 10 'set/define the operating pins for the LCD
'lcd 1, 1, " PHASE CONTROL BATTERY CHARGER VER 1.1"
'PAUSE 2000
PWM 2, 25000, 50            'PIN 26. HF trigger pulses for the SCR, gated by PIN 14
SETPIN 2, AIN               'Measured volts in. 0-3V3.'
SETPIN 3, AIN               'Measured Current in
SETPIN 4, AIN               'Set Voltage in
SETPIN 5, AIN               'Set Current in
PAUSE 500
PAUSE 500
SETPIN 15, CIN              'Trigger input for Oneshot.
SETPIN 16, CIN ', trigger   'PIN 16 connected to PIN 15 to capture trigger
SETPIN 17, INTH, OS_Reset   'out of sync
SETPIN 24, DOUT             'Heatbeat. Check prog is running
PIN(17) = 0                 'set value
oneshot 0,0,0               'reset oneshot
delay% = 9500               'set initial value for Oneshot
width% = 500                'set initial value for Oneshot
oneshot 25, delay%, width%  'Pin, delay & width
DV_set% = PIN(4) * 100      'set initial value for delay set
Delay% = 9500
sample_flag% = 1            'set flag for initial run through
  'Set Titles on LCD
LCD 1, 1, "VOLTS"
LCD 2, 1, " SET"
LCD 1, 13, "CURR"
LCD 2, 13, " SET"
LCD 1, 24, "MODE"
LCD 2, 24, "COND"

sample_flag% = 9                'initial value.
  DO                            'Operating period = 100mS. Measurements synched with 10th trigger input
    DO WHILE PIN(16) < 10       'wait for 10 I/P pulses (100mSec) to elapse
  LOOP
  SETPIN 16, CIN                        'reset counter to 0
  ' take measurements / readings
  PIN(24) = 1                           'toggle pin 24 to act as a prog heartbeat
  DV_Sample% = PIN(2) * 100             'Volts sample reading * scale. Also makes value an Integer eg 2.353
> 235
  DV_set%    = PIN(4) * 100             'Volts setting reading * scale
  V_diff%    = (DV_sample% - DV_set%)   'Difference between the two. Becomes the ERROR value
  IF V_diff% < -5 THEN                  'Detect if constant current or const volts mode
    CC_Flag = 1                         'Constant current
  ELSE
```

```
    CC_Flag = 0                          'Constant volts
  ENDIF
  DI_Sample%  = PIN(3) * 150             'Current sample reading * scale > integer
  DI_set%     = PIN(5) * 150             'Current setting reading * scale> integer
  I_diff%     = (DI_sample% - DI_set%)   'difference between the two. -ve if current high. ERROR Value
  IF DI_Set% < 2 THEN                    'check if Current Pot is near 0.
    delay% = 9500                        'max delay, set minimum current
    Mode$ = " OFF  "                    'Limit mode indication
  ELSEIF CC_Flag = 1 OR (DI_Sample% > DI_Set%)  THEN  'CC mode or current higher than setting
    delay% = delay% + (I_Diff% * 5)      'if so, adjust the delay. Mult factor affects response time &
stability.
    IF Delay% > 851 THEN
      Mode$ = "C_LIM "                  'Limit mode indication
    ELSE
      Mode$ = " MAX  "                  'Max delay. Charger maxxed out.
    ENDIF
  ELSE
    delay% = delay% + (V_diff% * 20)     'Not I lim, check volts against set
    IF V_diff% > -2 THEN                 'check if actually voltage limiting
      Mode$ = "V_LIM "                  'Limit mode indication
    ELSE                                 'not volt limiting either
      Mode$ = " MAX  "                  'charger maxxed out. Can't supply req'd volts
    ENDIF
  ENDIF
  PIN(24) = 0                            'toggle pin 24 to act as a prog heartbeat
  delay% = max(850, min(9500, delay%))   'delay limited from ~ 0.5mS to 8.5mS. Figure in uSecs
  width% = 9850 - delay%                 'set pulse width (firing time). 9850 = 9850 uSec = 10mSec - Trig
Pulse width
  oneshot 25, delay%, width%             'set main O/P pulses on PIN 25
  sample_flag% =  sample_flag% + 1       'inc 100mSec counter

    'Values converted to Strings & displayed on LCD or sent to terminal O/P
  IF  sample_flag% => 10 THEN            '1 sec readings to screen & LCD
    P$ = STR$(DV_sample%, 3)             'Raw voltage return sample
    PRINT P$;
    P$ = STR$(DV_Set%, 5)                'Raw voltage at Set Pot
    PRINT P$ ;
    V% = DV_Sample% * 5                  'Calculated input volts
    PRINT STR$((V% / 100), 4, 1);        'print formatted volts
    LCD 1, 7 , STR$((V% / 100),2, 1)
    V1% = DV_set% * 5                    'Calculate Pot Set volts
    PRINT STR$((V1% / 100), 4, 1);       'print formatted volts for monitoring
    LCD 2, 7 ,STR$((V1% / 100), 2, 1)    'display value to LCD
    PRINT STR$(DI_sample%, 5);           'print raw sampled current
    PRINT STR$(DI_set%, 5);              'print raw set current
    I% = DI_Sample% * 50                 'convert measured current to mA & integer
    PRINT STR$((I% /1000), 4, 1);        'print measured current in amps
    LCD 1, 18, STR$((I% /1000), 2, 1)
    I1% = DI_set% * 50                   'convert set current to mA & integer
    PRINT STR$((I1%/ 1000), 4, 1);       'print set current in amps
    LCD 2, 18, STR$((I1%/ 1000), 2, 1)
    angle$ = STR$(100 - (((delay% -850)/ 8650)*100),4,0)
    PRINT " " + angle$;                  'Delay percentage.
    LCD 2, 28, angle$ +"%"
    PRINT " " + STR$(delay%, 3,0);
    PRINT "  " + Mode$                  'print limit status
    LCD 1, 29, Mode$
    sample_flag% = 0                     'clear flag
   ' print timer
  ENDIF

LOOP                                     'go repeat loop

SUB trigger
oneshot 25, delay%, width%               '"Delayed"  O/P on PIN 25
END SUB

SUB OS_Reset                             'triggered if Trig & 1 Shot out of sync
oneshot 0,0,0                            'reset oneshot
oneshot 25, delay%, width%               're-initialise oneshot.
END SUB

'
' subroutine oneshot
' outputs a positive going pulse after a pre-determined period after a trigger pulse is received
' will continute to respond to each trigger pulse received until the output is turned off.
' The trigger pulse must be applied to pin 15 of a 28-pin Micromite which must be previously set up as a
counting input
' This function takes as parameters the following
' pin number of the pin to output the pulse
' pause after trigger before the pulse in microseconds (maximum is 524288/CPU speed = 13.1mSec @ 40MHz
' duration of the pulse in microseconds (maximum is 524288/CPU speed = 13.1mSec @ 40MHz
' Use pin number = 0 to turn off the output
' WARNING - the program has no error checking. Pulse durations greater than specified above will be rounded
down modulus 524288/CPU speed
'
```

```
CSUB oneshot
  00000030
  'T1Int
  3C039D00 8C62008C 8C450014 3C04BF80 AC850620 8C440008 8C420004 AC820000
  8C63008C 8C62000C 2444FFFF 14400008 AC64000C 24030010 3C02BF88 AC431064
  3C02BF88 AC401068 3C02BF80 AC400600 03E00008 00000000
  'E2Int
  3C029D00 8C43008C 24040001 AC64000C 8C42008C 8C430010 3C02BF80 AC430620
  34038010 3C02BF80 AC430600 24020010 3C03BF88 AC621064 3C03BF88 AC621068
  03E00008 00000000
  'getFPC
  27BDFFF8 AFBF0004 00852023 03E42021 ACC40000 8FBF0004 03E00008 27BD0008
  'main
  27BDFFC8 AFBF0034 AFB50030 AFB4002C AFB30028 AFB20024 AFB10020 AFB0001C
  00808021 00A08821 00C09021 00002021 3C059D00 24A50104 27A60010 0411FFE8
  00000000 8E530000 3C029D00 8C420000 8C520000 8E020000 8E030004 00431025
  1440000E 8E340000 24030010 3C02BF88 AC431064 3C02BF88 AC401068 3C02BF80
  AC400600 3C029D00 8C4300CC AC600000 8C4200AC 1000004B AC400000 3C029D00
  8C4200CC 8C420000 1440002A 3C03BF80 24030010 3C02BF88 AC431064 3C02BF88
  AC401068 3C119D00 8E220010 8E040000 24050008 0040F809 00003021 8E35008C
  8E220024 8E040000 0040F809 24050007 AEA20008 8E35008C 8E220028 0040F809
  8E040000 24030001 00431004 AEA20004 8E22001C 8E040000 0040F809 24050005
  8FA40010 8E2200CC 3C039D00 24630058 00641821 AC430000 8FA40010 8E2200AC
  3C039D00 24630000 00641821 10000004 AC430000 8C620600 1440FFFE 00000000
  3C029D00 8C43008C 24040001 AC64000C 001290C2 3C03000F 24634240 0243001A
  006001F4 00009012 8C43008C 7254A002 AC740010 8C42008C 72539002 AC520014
  2403001C 3C02BF88 AC4310A4 24030004 3C02BF88 AC4310A8 24030010 3C02BF88
  AC431034 8FBF0034 8FB50030 8FB4002C 8FB30028 8FB20024 8FB10020 8FB0001C
  03E00008 27BD0038
END CSUB
```