```
''****************************
''*  VGA Driver v1.1         *
''*  (C) 2006 Parallax, Inc.  *
''****************************
''
'' v1.0 - 01 May 2006 - original version
'' v1.1 - 15 May 2006 - pixel tile size can now be 16 x 32 to enable mor
''                      character displays utilizing the internal font -

CON

  paramcount    = 21
  colortable    = $180  'start of colortable inside cog


VAR

  long  cog


PUB start(vgaptr) : okay

'' Start VGA driver - starts a cog
'' returns false if no cog available
''
''   vgaptr = pointer to VGA parameters
''
'' the driver reloads all parameters each refresh,
'' allowing you to make live changes to them

  stop
  okay := cog := cognew(@entry, vgaptr) + 1


PUB stop

'' Stop VGA driver - frees a cog

  if cog
    cogstop(cog~ - 1)


DAT

'*********************************
'* Assembly language VGA driver *
'*********************************

                        org
```

```
'
'
' Entry
'
entry                   mov     taskptr,#tasks          'reset tasks

                        mov     x,#8                    'perform task se
:init                   jmpret  taskret,taskptr
                        djnz    x,#:init
'
'
' Superfield
'
superfield              mov     hv,hvbase               'set hv

                        mov     interlace,#0            'reset interlace

                        test    _mode,#%0100    wz      'get interlace i
'
'
' Field
'
field                   wrlong  visible,par             'set status to v

                        tjz     vb,#:nobl               'do any visible
                        mov     x,vb
                        movd    bcolor,#colortable
                        call    #blank_line
:nobl
                        mov     screen,_screen          'point to first
                        mov     y,_vt                   'set vertical ti
:line                   mov     vx,_vx                  'set vertical ex
:vert   if_nz           xor     interlace,#1            'interlace skip?
        if_nz           tjz     interlace,#:skip

                        tjz     hb,#:nobp               'do any visible
                        mov     vscl,hb
                        waitvid colortable,#0
:nobp
                        mov     x,_ht                   'set horizontal
                        mov     vscl,hx                 'set horizontal

:tile                   rdword  tile,screen             'read tile
                        add     tile,line               'set pointer bit
                        rol     tile,#6                 'read tile pixel
                        rdlong  pixels,tile             '(8 clocks betwe
                        shr     tile,#10+6              'set tile colors
                        movd    :color,tile
                        add     screen,#2               'point to next t
```

```
:color                  waitvid colortable,pixels       'pass colors and
                        djnz    x,#:tile                'another tile?

                        sub     screen,hc2x             'repoint to firs

                        tjz     hf,#:nofp               'do any visible
                        mov     vscl,hf
                        waitvid colortable,#0
:nofp
                        mov     x,#1                    'do hsync
                        call    #blank_hsync            '(x=0)

:skip                   djnz    vx,#:vert               'vertical expand
                        ror     line,linerot            'set next line
                        add     line,lineadd    wc
                        rol     line,linerot
        if_nc           jmp     #:line
                        add     screen,hc2x             'point to first
                        djnz    y,#:line        wc      'another tile li

                        tjz     vf,#:nofl               'do any visible
                        mov     x,vf
                        movd    bcolor,#colortable
                        call    #blank_line
:nofl
        if_nz           xor     interlace,#1    wc,wz   'get interlace a

        if_z            wrlong  invisible,par           'unless interlac

                        mov     taskptr,#tasks          'reset tasks

                        addx    x,_vf           wc      'do invisible fr
                        call    #blank_line

                        mov     x,_vs                   'do vsync lines
                        call    #blank_vsync

                        mov     x,_vb                   'do invisible ba
                        call    #blank_vsync

        if_nz           jmp     #field                  'if interlace an
                        jmp     #superfield             'else, new super
'
'
' Blank line(s)
'
blank_vsync             cmp     interlace,#2    wc      'vsync (c=1)

blank_line              mov     vscl,h1                 'blank line or v
```

```
        if_nc           add     vscl,h2
        if_c_and_nz     xor     hv,#%01
        if_c            waitvid hv,#0
        if_c            mov     vscl,h2                 'blank line or v
        if_c_and_z      xor     hv,#%01
bcolor                  waitvid hv,#0

        if_nc           jmpret  taskret,taskptr         'call task secti

blank_hsync             mov     vscl,_hf                'hsync, do invis
                        waitvid hv,#0

                        mov     vscl,_hs                'do invisble syn
                        xor     hv,#%10
                        waitvid hv,#0

                        mov     vscl,_hb                'do invisible ba
                        xor     hv,#%10
                        waitvid hv,#0

                        djnz    x,#blank_line   wc      '(c=0)

                        movd    bcolor,#hv
blank_hsync_ret
blank_line_ret
blank_vsync_ret         ret
'
'
' Tasks - performed in sections during invisible back porch lines
'
tasks                   mov     t1,par                  'load parameters
                        movd    :par,#_enable           '(skip _status)
                        mov     t2,#paramcount - 1
:load                   add     t1,#4
:par                    rdlong  0,t1
                        add     :par,d0
                        djnz    t2,#:load               '+164

                        mov     t1,#2                   'set video pins
                        shl     t1,_pins                '(if video disab
                        sub     t1,#1
                        test    _pins,#$20      wc
                        and     _pins,#$38
                        shr     t1,_pins
                        movs    vcfg,t1
                        shl     t1,_pins
                        shr     _pins,#3
                        movd    vcfg,_pins
        if_nc           mov     dira,t1
```

```
        if_nc           mov     dirb,#0
        if_c            mov     dira,#0
        if_c            mov     dirb,t1                 '+14

                        tjz     _enable,#disabled       '+2, disabled?

                        jmpret  taskptr,taskret         '+1=181, break a

                        rdlong  t1,#0                   'make sure CLKFR
                        shr     t1,#1
                        cmp     t1,m8           wc
        if_c            jmp     #disabled               '+8

                        min     _rate,pllmin            'limit _rate to
                        max     _rate,pllmax            '+2

                        mov     t1,#%00001_011          'set ctra config
:max                    cmp     m8,_rate        wc      'adjust rate to
        if_c            shr     _rate,#1                '(vco will be wi
        if_c            add     t1,#%00000_001
        if_c            jmp     #:max
:min                    cmp     _rate,m4        wc
        if_c            shl     _rate,#1
        if_c            sub     x,#%00000_001
        if_c            jmp     #:min
                        movi    ctra,t1                 '+22

                        rdlong  t1,#0                   'divide _rate/CL
                        mov     hvbase,#32+1
:div                    cmpsub  _rate,t1        wc
                        rcl     t2,#1
                        shl     _rate,#1
                        djnz    hvbase,#:div            '(hvbase=0)
                        mov     frqa,t2                 '+136

                        test    _mode,#%0001    wc      'make hvbase
                        muxnc   hvbase,vmask
                        test    _mode,#%0010    wc
                        muxnc   hvbase,hmask            '+4

                        jmpret  taskptr,taskret         '+1=173, break a

                        mov     hx,_hx                  'compute horizon
                        shl     hx,#8
                        or      hx,_hx
                        shl     hx,#4

                        mov     hc2x,_ht
                        shl     hc2x,#1
```

```
                        mov     h1,_hd
                        neg     h2,_hf
                        sub     h2,_hs
                        sub     h2,_hb
                        sub     h1,h2
                        shr     h1,#1           wc
                        addx    h2,h1

                        mov     t1,_ht
                        mov     t2,_hx
                        call    #multiply
                        mov     hf,_hd
                        sub     hf,t1
                        shr     hf,#1           wc
                        mov     hb,_ho
                        addx    hb,hf
                        sub     hf,_ho                  '+59

                        mov     t1,_vt                  'compute vertica
                        mov     t2,_vx
                        call    #multiply
                        test    _mode,#%1000    wc      'consider tile s
                        muxc    linerot,#1
                        mov     lineadd,lineinc
        if_c            shr     lineadd,#1
        if_c            shl     t1,#1
                        test    _mode,#%0100    wc      'consider interl
        if_c            shr     t1,#1
                        mov     vf,_vd
                        sub     vf,t1
                        shr     vf,#1           wc
                        neg     vb,_vo
                        addx    vb,vf
                        add     vf,_vo                  '+53

                        movi    vcfg,#%01100_000        '+1, set video c

:colors                 jmpret  taskptr,taskret         '+1=114/160, bre

                        mov     t1,#13                  'load next 13 co
:loop                   mov     t2,:color               '5 times = 65 (a
                        shr     t2,#9-2
                        and     t2,#$FC
                        add     t2,_colors
                        rdlong  t2,t2
                        and     t2,colormask
                        or      t2,hvbase
:color                  mov     colortable,t2
```

```
                        add     :color,d0
                        andn    :color,d6
                        djnz    t1,#:loop               '+158

                        jmp     #:colors                '+1, keep loadin
'
'
' Multiply t1 * t2 * 16 (t1, t2 = bytes)
'
multiply                shl     t2,#8+4-1

                        mov     tile,#8
:loop                   shr     t1,#1           wc
        if_c            add     t1,t2
                        djnz    tile,#:loop

multiply_ret            ret                             '+37
'
'
' Disabled - reset status, nap ~4ms, try again
'
disabled                mov     ctra,#0                 'reset ctra
                        mov     vcfg,#0                 'reset video

                        wrlong  outa,par                'set status to d

                        rdlong  t1,#0                   'get CLKFREQ
                        shr     t1,#8                   'nap for ~4ms
                        min     t1,#3
                        add     t1,cnt
                        waitcnt t1,#0

                        jmp     #entry                  'reload paramete
'
'
' Initialized data
'
pllmin                  long    500_000                 'pll lowest outp
pllmax                  long    128_000_000             'pll highest out
m8                      long    8_000_000               '*16 = 128MHz (p
m4                      long    4_000_000               '*16 = 64MHz (pl
d0                      long    1 << 9 << 0
d6                      long    1 << 9 << 6
invisible               long    1
visible                 long    2
line                    long    $00060000
lineinc                 long    $10000000
linerot                 long    0
vmask                   long    $01010101
```

```
hmask                   long    $02020202
colormask               long    $FCFCFCFC
'
'
' Uninitialized data
'
taskptr                 res     1                       'tasks
taskret                 res     1
t1                      res     1
t2                      res     1

x                       res     1                       'display
y                       res     1
hf                      res     1
hb                      res     1
vf                      res     1
vb                      res     1
hx                      res     1
vx                      res     1
hc2x                    res     1
screen                  res     1
tile                    res     1
pixels                  res     1
lineadd                 res     1
interlace               res     1
hv                      res     1
hvbase                  res     1
h1                      res     1
h2                      res     1
'
'
' Parameter buffer
'
_enable                 res     1       '0/non-0        read-only
_pins                   res     1       '%pppttt        read-only
_mode                   res     1       '%tihv          read-only
_screen                 res     1       '@word          read-only
_colors                 res     1       '@long          read-only
_ht                     res     1       '1+             read-only
_vt                     res     1       '1+             read-only
_hx                     res     1       '1+             read-only
_vx                     res     1       '1+             read-only
_ho                     res     1       '0+-            read-only
_vo                     res     1       '0+-            read-only
_hd                     res     1       '1+             read-only
_hf                     res     1       '1+             read-only
_hs                     res     1       '1+             read-only
_hb                     res     1       '1+             read-only
_vd                     res     1       '1+             read-only
```

```
_vf                     res     1       '1+             read-only
_vs                     res     1       '1+             read-only
_vb                     res     1       '2+             read-only
_rate                   res     1       '500_000+       read-only

                        fit     colortable              'fit underneath


''
''  ___
''VAR                   'VGA parameters - 21 contiguous longs
''
''  long  vga_status    '0/1/2 = off/visible/invisible      read-only
''  long  vga_enable    '0/non-0 = off/on                   write-only
''  long  vga_pins      '%pppttt = pins                     write-only
''  long  vga_mode      '%tihv = tile,interlace,hpol,vpol   write-only
''  long  vga_screen    'pointer to screen (words)          write-only
''  long  vga_colors    'pointer to colors (longs)          write-only
''  long  vga_ht        'horizontal tiles                   write-only
''  long  vga_vt        'vertical tiles                     write-only
''  long  vga_hx        'horizontal tile expansion          write-only
''  long  vga_vx        'vertical tile expansion            write-only
''  long  vga_ho        'horizontal offset                  write-only
''  long  vga_vo        'vertical offset                    write-only
''  long  vga_hd        'horizontal display ticks           write-only
''  long  vga_hf        'horizontal front porch ticks       write-only
''  long  vga_hs        'horizontal sync ticks              write-only
''  long  vga_hb        'horizontal back porch ticks        write-only
''  long  vga_vd        'vertical display lines             write-only
''  long  vga_vf        'vertical front porch lines         write-only
''  long  vga_vs        'vertical sync lines                write-only
''  long  vga_vb        'vertical back porch lines          write-only
''  long  vga_rate      'tick rate (Hz)                     write-only
''
''The preceding VAR section may be copied into your code.
''After setting variables, do start(@vga_status) to start driver.
''
''All parameters are reloaded each superframe, allowing you to make live
''changes. To minimize flicker, correlate changes with vga_status.
''
''Experimentation may be required to optimize some parameters.
''
''Parameter descriptions:
''
''  ___________
''  vga_status
''
''    driver sets this to indicate status:
''      0: driver disabled (vga_enable = 0 or CLKFREQ < 16MHz)
''      1: currently outputting invisible sync data
```

```
''      2: currently outputting visible screen data
''
''    ___________
''    vga_enable
''
''         0: disable (pins will be driven low, reduces power)
''      non-0: enable
''
''    _________
''    vga_pins
''
''      bits 5..3 select pin group:
''        %000: pins 7..0
''        %001: pins 15..8
''        %010: pins 23..16
''        %011: pins 31..24
''        %100: pins 39..32
''        %101: pins 47..40
''        %110: pins 55..48
''        %111: pins 63..56
''
''      bits 2..0 select top pin within group
''      for example: %01111 (15) will use pins %01000-%01111 (8-15)
''
''    _________
''    vga_mode
''
''      bit 3 selects between 16x16 and 16x32 pixel tiles:
''        0: 16x16 pixel tiles (tileheight = 16)
''        1: 16x32 pixel tiles (tileheight = 32)
''
''      bit 2 controls interlace:
''        0: progressive scan (less flicker, good for motion, required for
''        1: interlaced scan (allows you to double vga_vt, good for text)
''
''      bits 1 and 0 select horizontal and vertical sync polarity, respect
''        0: active low
''        1: active high
''
''    ___________
''    vga_screen
''
''      pointer to words which define screen contents (left-to-right, top-
''        number of words must be vga_ht * vga_vt
''        each word has two bitfields: a 6-bit colorset ptr and a 10-bit p
''          bits 15..10: select the colorset* for the associated pixel til
''          bits 9..0: select the pixelgroup** address %ppppppppppcccc00 (
''
''         * colorsets are longs which each define four 8-bit colors
''
''        ** pixelgroups are <tileheight> longs which define (left-to-righ
''           (four color) pixels that make up a 16x16 or a 16x32 pixel til
''
''    ___________
```

```
''  vga_colors
''
''    pointer to longs which define colorsets
''      number of longs must be 1..64
''      each long has four 8-bit fields which define colors for 2-bit (f
''      first long's bottom color is also used as the screen background
''      8-bit color fields are as follows:
''        bits 7..2: actual state of pins 7..2 within pin group*
''        bits 1..0: don't care (used within driver for hsync and vsync)
''
''    * it is suggested that:
''        bits/pins 7..6 are used for red
''        bits/pins 5..4 are used for green
''        bits/pins 3..2 are used for blue
''      for each bit/pin set, sum 240 and 470-ohm resistors to form 75-o
''      connect signal sets to RED, GREEN, and BLUE on VGA connector
''      always connect group pin 1 to HSYNC on VGA connector via 240-ohm
''      always connect group pin 0 to VSYNC on VGA connector via 240-ohm
''
''  _______
''  vga_ht
''
''    horizontal number of pixel tiles - must be at least 1
''
''  _______
''  vga_vt
''
''    vertical number of pixel tiles - must be at least 1
''
''  _______
''  vga_hx
''
''    horizontal tile expansion factor - must be at least 1
''
''    make sure 16 * vga_ht * vga_hx + ||vga_ho is equal to or at least
''
''  _______
''  vga_vx
''
''    vertical tile expansion factor - must be at least 1
''
''    make sure <tileheight> * vga_vt * vga_vx + ||vga_vo does not excee
''      (for interlace, use <tileheight> / 2 * vga_vt * vga_vx + ||vga_v
''
''  _______
''  vga_ho
''
''    horizontal offset in ticks - pos/neg value (0 recommended)
''    shifts the display right/left
''
''  _______
''  vga_vo
''
''    vertical offset in lines - pos/neg value (0 recommended)
''    shifts the display up/down
```

```
''
''   _______
''   vga_hd
''
''     horizontal display ticks
''
''   _______
''   vga_hf
''
''     horizontal front porch ticks
''
''   _______
''   vga_hs
''
''     horizontal sync ticks
''
''   _______
''   vga_hb
''
''     horizontal back porch ticks
''
''   _______
''   vga_vd
''
''     vertical display lines
''
''   _______
''   vga_vf
''
''     vertical front porch lines
''
''   _______
''   vga_vs
''
''     vertical sync lines
''
''   _______
''   vga_vb
''
''     vertical back porch lines
''
''   _________
''   vga_rate
''
''     tick rate in Hz
''
''     driver will limit value to be within 500KHz and 128MHz
''     pixel rate (vga_rate / vga_hx) should be no more than CLKFREQ / 4
```