PropGFX Programming Reference Guide

The PropGFX is a 40pin DIP device, for use in your electronics projects, to give it colour TV output, with bitmap or character map displays, with sprites.
(including some retro style modes like Sinclair Spectrum, and Amstrad CPC, and Commodore 64 modes if you wanted to do some retro style games.)

It is capable of both PAL and NTSC and VGA displays.
Based around a Propeller Microcontroller from Parallax (http://www.parallax.com) which is a Microcontroller with 32KB of ram and has 8 “cogs”, each cog is a 32bit processor that has 2KB of it's own program ram, which has access to the main 32KB of RAM.
The PropGFX has 1 Comms Cog, 1 Display Driver Cog, and 2,4,5,6 Render Cogs, and the remaining cogs are allocated as Poly Cogs.

Right first off, lets tell you how to communicate with the **PropGFX**.

There are two ways to send and receive data to and from the **PropGFX**.

Comms Mode 0 is **Serial Mode**.
Comms Mode 1 is **Databus Mode**, which has two control pins, and 8 data pins.

The **PropGFX** is set initially to Serial Mode.

Serial Mode is set initially to 9600 baud, and can go up to 1.7Mbps.
Databus Mode has two control pins and 8 data pins.

Sending and receiving data to and from **PropGFX** in **Serial Mode...**

This can be done with usual rs232 style communications :)
The **PropGFX** starts initially in Serial Mode, but you can swap it to DataBus without needing to use serial comms, by Sending a dummy byte in **Databus Mode** (any value other than 0 and "J") to the **PropGFX**.

Sending and receiving data to and from **PropGFX** in **Databus Mode...**

Sending a Data Byte to **PropGFX**

Wait for RX to be Low
Set your Data on the DataBus
Set TX to High
Wait for RX to be High
Set TX and DataBus to Low

Reading a Data Byte from **PropGFX**

Set TX to High
Wait for RX to be High
Get your Data from the DataBus

PropGFX Programming Reference Guide

Set TX to Low
Wait for RX to be Low
The Comms code has special command strings it looks for when reading Data Bytes sent to it, so if it's host was reset or something during comms, but it wasn't, it can (99.9% of the time) distinguish between data, and commands :).

These command strings are as follows :-

Upload buffer to **PropGFX** RAM

"J","B","U","P",**dstAdrLo**,**dstAdrHi**,**lenLo**,**lenHi**,followed by "**len**" bytes of data

This will allow you to upload your data to RAM at **dstAdr** on the **PropGFX**

Wait for Vsync

"J","B","V","S"

This will wait for Vertical Sync of the display, so you can use this for syncing your graphics (frame swap etc.)

Start a Cog with a new program from **PropGFX** RAM

"J","B","C","G",**cogInitInfoLoWordLo**,**cogInitInfoLoWordHi**,**cogInitInfoHiWordLo**,**cogInitInfoHiWordHi**

This will reset one of the cogs. (Not to be used without good knowledge of Propeller Microcontroller)

Debug LED on/off

"J","B","D","B",**OnOff**

This will turn the **PropGFX** debug LED on/off depending on OnOff value (Odd value (bit 0 set) = On, Even value (bit 0 clear) = Off)

Download buffer from **PropGFX** RAM

"J","B","D","N",**srcAdrLo**,**srcAdrHi**,**lenLo**,**lenHi**, you then have to Read "**len**" bytes of data

This will allow you to download your data from RAM at **srcAdr** on the **PropGFX**

Change Comms Mode (to Serial or DataBus)

PropGFX Programming Reference Guide

"J","B","I","O",**newmode**

This will allow you to change comms mode (to and from databus or serial) (Databus = 1 , Serial = 0)

Get ID Version

"J","B","I","D", then Read 4 bytes of data

This is how you can identify which PropGFX you're working with (**PropGFX** Lite = "L","0","0","1" , **PropGFX** VGA = "V","0","0","1")

PropGFX EEPROM Read to **PropGFX** RAM

"J","B","E","R",**srcAdrLo**,**srcAdrHi**,**dstAdrLo**,**dstAdrHi**,**lenLo**,**lenHi**, you then have to Read "**len**" bytes of data

This will copy "**len**" bytes from **PropGFX** EEPROM **srcAdr** to PropGFX RAM **dstAdr** (and send an OK on completion in serial mode)

PropGFX EEPROM Write from **PropGFX** RAM

"J","B","E","W",**srcAdrLo**,**srcAdrHi**,**dstAdrLo**,**dstAdrHi**,**lenLo**,**lenHi**, you then have to Write "len" bytes of data

This will copy "**len**" bytes from **PropGFX** RAM **srcAdr** to **PropGFX** EEPROM **dstAdr** (and send an OK on completion in serial mode)

Set Baud Rate (for serial mode)

"J","B","B","D",**baudLo**,**baudHi**, you then have to Read an OK byte of "0"

This is how to set a new baud rate for PropGFX in Serial Mode, precalculate baud value **baud** = 96_000_000/baud (100_000_000/baud for VGA)

Reset PropGX

"J","B","R","S",**resettype**

This will reset the **PropGFX** depending on **resettype** (0 = Will just Reset the TV and Render Cogs to values in their system vars, 1 = Will do a full cold reset) (and send an OK on completion in serial mode except in cold reset)

PropGFX Programming Reference Guide

Send a Go command to Poly Cog

"J","B","P","L"

This will set a Go command in the 3D cog stats, as you can't write longs only bytes with JBUP (and send an OK on completion in serial mode)

Fill Words in **PropGFX** RAM

"J","B","F","W",**srcAdrLo**,**srcAdrHi**,**valLo**,**valHi**,**lenLo**,**lenHi**

This will fill "len" words of **PropGFX** RAM at "**srcAdr**" with "**val**" (and send an OK on completion in serial mode)

Move Words in **PropGFX** RAM

"J","B","M","W",**srcAdrLo**,**srcAdrHi**,**dstAdrLo**,**dstAdrHi**,**lenLo**,**lenHi**

This will move "**len**" words of **PropGFX** RAM at "**srcAdr**" to **PropGFX** RAM at "**dstAdr**" (incremental direction only) (and send an OK on completion in serial mode)

PropGFX RAM layout and Defines

```
PROPGFX_COGNUM          = $18          ' LONG - ( DO NOT CHANGE THIS! )

PROPGFX_STATUS          = $1c          ' LONG - Status flag ( used by Parallax driver. )

PROPGFX_ENABLE          = $20          ' LONG - Enable TV.

PROPGFX_TVPINS          = $24          ' LONG - TVPins ( DO NOT CHANGE THIS! )

PROPGFX_TVMODE          = $28          ' LONG - 0 = NTSC, 1 = PAL, 2 = NTSC interlaced mode, 3 = PAL interlaced mode.
PROPGFX_NTSC            = 0 ' use this to set region mode for NTSC
PROPGFX_PAL             = 1 ' use this to set region mode to PAL

PROPGFX_XTILES          = $2c          ' LONG - X number of horizontal characters in the display.

PROPGFX_YTILES          = $30          ' LONG - Y number of vertical characters in the display.

PROPGFX_SCALEX          = $34          ' LONG - SCALEX alters the X scale of the
```

pixels in the Horizontal axis.

PROPGFX_SCALEY = $38 ' LONG - (DO NOT CHANGE THIS it's set to 1)

PROPGFX_OFFSETX = $3c ' LONG - X screen offset.

PROPGFX_OFFSETY = $40 ' LONG - Y screen offset.

PROPGFX_BROADCAST = $44 ' LONG - (DO NOT CHANGE THIS!)

PROPGFX_AURALCOG = $48 ' LONG - (DO NOT CHANGE THIS!)

PROPGFX_LINEBUF = $4c ' LONG - (DO NOT CHANGE THIS!) Starting point of the display line render buffers (352 bytes per cog used for rendering scanlines.)
PROPGFX_PIXEL_BUFF_BASE = $00d4 ' where the scanlines are generated
PROPGFX_PIXEL_BUFF_END2 = $039c ' end of the second scanline render cog's buffer. (can't display using one cog it's not really fast enough.)
PROPGFX_PIXEL_BUFF_END3 = $04fc ' end of the third scanline render cog's buffer.
PROPGFX_PIXEL_BUFF_END4 = $065c ' end of the fourth scanline render cogs' buffer.
PROPGFX_PIXEL_BUFF_END5 = $07bc ' end of the fifth scanline render cogs' buffer.
PROPGFX_PIXEL_BUFF_END6 = $091c ' end of the sixth scanline render cogs' buffer.

PROPGFX_BORDERPTR = $50 ' WORD - (DO NOT CHANGE THIS!) Pointer to where to get the border colour from (fixed to $A4) for previous versions.

PROPGFX_DOSPLITMODE = $52 ' BYTE - DoSplitMode set this byte to set the split mode type, and set bit 7 to activate Split screen mode (on all but Parallax mode displays.)
PROPGFX_SET_SPLIT_MODE = $80
PROPGFX_2BIT_CHAR_MODE = 1
PROPGFX_4BIT_CHAR_MODE = 2
PROPGFX_8BIT_BIG_BITMAP_MODE = 3
PROPGFX_1BIT_BITMAP_MODE = 4
PROPGFX_2BIT_BITMAP_MODE = 5
PROPGFX_4BIT_BITMAP_MODE = 6
PROPGFX_8BIT_BITMAP_MODE = 7
PROPGFX_SPECCY_MODE = 0
PROPGFX_SPECCY_HICOLOUR = 1
PROPGFX_SPECCY_CHARMAP = 2
PROPGFX_C64_CHARMAP_MODE_1BIT = 0
PROPGFX_C64_CHARMAP_MODE_2BIT = 1
PROPGFX_C64_BITMAP_MODE_1BIT = 2
PROPGFX_C64_BITMAP_MODE_2BIT = 3
PROPGFX_AMS_BITMAP_MODE_2BIT = 0
PROPGFX_AMS_BITMAP_MODE_4BIT = 1
PROPGFX_AMS_CHARMAP_MODE_2BIT = 2
PROPGFX_AMS_CHARMAP_MODE_4BIT = 3

PropGFX Programming Reference Guide

PROPGFX_SPLITY = $53 ' BYTE - SplitY scanline.

PROPGFX_NEXTLNPTR = $54 ' WORD - (DO NOT CHANGE THIS!)

PROPGFX_SPLIT_XSCR = $56 ' BYTE - SPLIT_XSCR X pixel scroll offset (0-7) for after the split.

PROPGFX_SPLIT_YSCR = $57 ' BYTE - SPLIT_YSCR Y pixel scroll offset (0-7) for after the split.

PROPGFX_MODE = $58 ' WORD - Mode sets which type of graphics mode you want to display dependant on which render drivers you have chosen (on all but Parallax mode displays.)
PROPGFX_MODE_XSCROLLERS_ENABLE = 8

PROPGFX_BITMAPPTR = $5a ' WORD - BITMAPPTR pointer to the first byte of the bitmap dispay.
PROPGFX_SPECCY_ATTR_BASE = $2000+$1800
PROPGFX_BITMAP_BASE = $2000

PROPGFX_VSYNCVAL = $5c ' WORD - VSYNCVAL set to 0, 1, or 2 by the TV driver,0 = set to 0 on start of display, 1 = set to 1 on VSYNC, 2 = set to 2 at Start line of bottom border.

PROPGFX_CHRMAPPTR = $60 ' WORD - CHRMAPPTR pointer to the first tile in the charmap displays.
PROPGFX_CHRMAP_BASE = $1800

PROPGFX_CHRMAPPTR_SPLIT = $62 ' WORD - CHRMAPPTR_SPLIT pointer to the first tile in the charmap displays after the split screen (in case you want to have a different font for the split.

PROPGFX_CHRSETPTR = $64 ' WORD - CHRSETPTR pointer to the first character in the charmap displays.
PROPGFX_CHRSET_BASE = $2000

PROPGFX_CHRSETPTR_SPLIT = $66 ' WORD - CHRSETPTR_SPLIT pointer to the first character in the charmap displays after the split screen (in case you want to have a different font for the split.

PROPGFX_CHRPALPTR = $68 ' WORD - CHRPALPTR pointer to the background palette.
PROPGFX_CHRPAL_BASE = $1600

PROPGFX_CHRPALPTR_SPLIT = $6a ' WORD - CHRPALPTR_SPLIT pointer to the background palette after split.

PROPGFX_COGCOUNT = $6c ' WORD - COGCOUNT number of scanline

rendering cogs, when reseting mode etc. valid numbers are 2,4,5,6, as the remainder from 6 (4,2,1,0) is used for poly rendering cogs.
PROPGFX_0_3D_COGS = 6 ' this is to set the amount of scanline renderers, (6 = 0 left for POLY cogs)
PROPGFX_1_3D_COG = 5 ' this is to set the amount of scanline renderers, (5 = 1 left for POLY cogs)
PROPGFX_1_3D_COGS = 5 ' this is to set the amount of scanline renderers, (5 = 1 left for POLY cogs) added this one incase people leave the trailing S
PROPGFX_2_3D_COGS = 4 ' this is to set the amount of scanline renderers, (4 = 2 left for POLY cogs)
PROPGFX_4_3D_COGS = 2 ' this is to set the amount of scanline renderers, (2 = 4 left for POLY cogs)
PROPGFX_0_SCROLL_COGS = 6
PROPGFX_1_SCROLL_COG = 5
PROPGFX_1_SCROLL_COGS = 5

PROPGFX_XSCROLLER = $70 ' BYTE - XSCROLLER 32 bytes one for each character line depending on multiscroll or single scroll value for whole display area.

PROPGFX_YSCROLLER = $71 ' BYTE - YSCROLLER when not using multiple X scrolling, this byte is the Y scroll value.

PROPGFX_SPRITECOUNT = $90 ' WORD - SPRITECOUNT ((actually a byte) max at around 128 just don't put them all around the same line) max per scanline depends on mode and cogcount

PROPGFX_SPRITECOUNT_SPLIT = $92 ' BYTE - SPRITECOUNT_SPLIT same as above, but for after the split.

PROPGFX_SPRITECTRL = $93 ' BYTE - SPRITECTRL uses 4 colour sprites unless you set bit 7 for 16colour sprites.

PROPGFX_SPRLSTPTR = $94 ' WORD - SPRLSTPTR pointer to sprite list
PROPGFX_SPRITE_LIST_BASE = $13c0

PROPGFX_SPRLSTPTR_SPLIT = $96 ' WORD - SPRLSTPTR_SPLIT pointer to sprite list after split.

PROPGFX_SPRPALPTR = $98 ' WORD - SPRPALPTR pointer to sprite palette.
PROPGFX_SPRPAL_BASE = $1700

PROPGFX_SPRPALPTR_SPLIT = $9a ' WORD - SPRPALPTR_SPLIT pointer to sprite palette after split.

PROPGFX_SPRCHRPTR = $9c ' WORD - SPRCHRPTR pointer to sprite character set (sprites are organised in a font, sprite 0 has character, 0,1, then 32,33, so it looks right on your sprite bmp)

PropGFX Programming Reference Guide

```
PROPGFX_SPRSET_BASE          = $5000
PROPGFX_SPRSET_BASE2         = $4000 'use for more sprite base when not in bitmap mode
PROPGFX_SPRSET_BASE_AMS      = $6000

PROPGFX_SPRCHRPTR_SPLIT      = $9e          ' WORD - SPRCHRPTR_SPLIT pointer to sprite character set after split.

PROPGFX_NEXTLINE             = $a0          ' LONG - ( DO NOT CHANGE THIS! )

PROPGFX_BORDERCOL            = $a4          ' WORD - ( actually a byte ) screen border colour using propeller palette.

PROPGFX_BITMAPPTR_SPLIT      = $a6          ' WORD - BITMAPPTR_SPLIT pointer to bitmap after the split.

PROPGFX_3DLISTPTR            = $a8          ' LONG - 3DLISTPTR pointer to poly draw list
PROPGFX_POLY_BUFF_BASE_BIG   = $680
PROPGFX_POLY_BUFF_BASE       = $1000

PROPGFX_3DSTAT1              = $ac          ' LONG - ( DO NOT CHANGE THIS! )

PROPGFX_3DSTAT2              = $b0          ' LONG - ( DO NOT CHANGE THIS! )

PROPGFX_DRIVEROFFSET         = $b4          ' LONG - ( DO NOT CHANGE THIS! )

PROPGFX_ID                   = $b8          ' LONG - ( DO NOT CHANGE THIS! )

PROPGFX_COMMSCOGPTR          = $bc          ' LONG - ( DO NOT CHANGE THIS! )
PROPGFX_COMMS_CODE           = $801a

PROPGFX_TVCOGPTR             = $c0          ' LONG - TVCOGPTR pointer to where to get cog code for tvdriver
PROPGFX_MAIN_TV_CODE         = $801c
PROPGFX_PARALLAX_TV_CODE     = $801e

PROPGFX_TILECOGPTR           = $c4          ' LONG - TILECOGPTR pointer to where to get cog code for tiledriver
PROPGFX_MAIN_TILE_CODE       = $8020
PROPGFX_SPECCY_TILE_CODE     = $8022
PROPGFX_C64_TILE_CODE        = $8024
PROPGFX_AMS_TILE_CODE        = $8026
PROPGFX_STOP_COG_DEAD_CODE   = $803e

PROPGFX_3DCOGPTR             = $c8          ' LONG - 3DCOGPTR pointer to where to get cog code for polydriver
PROPGFX_1BIT_3D_CODE         = $8028
PROPGFX_2BIT_3D_CODE         = $802a
PROPGFX_4BIT_3D_CODE         = $802c
```

PropGFX Programming Reference Guide

```
PROPGFX_8BIT_3D_CODE          = $802e
PROPGFX_1BIT_LINE_CODE        = $8030
PROPGFX_2BIT_LINE_CODE        = $8032
PROPGFX_4BIT_LINE_CODE        = $8034
PROPGFX_8BIT_LINE_CODE        = $8036
PROPGFX_SCROLLER_COG_CODE     = $8038
PROPGFX_STOP_COG_CODE         = $803a



'ResetLite options
PROPGFX_RESET_TILE            = 0 ' this one resets the Tile + 3D drivers only, and doesn't reset the TV driver or the Comms driver
PROPGFX_RESET_TV              = 1 ' this one resets the TV + Tile + 3D drivers, and doesn't reset the Comms driver
PROPGFX_RESET_FULL            = 2 ' this is like a cold reset, it does a full reboot.

'Triangle offsets ( in words )
PROPGFX_POLYC                 = 0
PROPGFX_POLYX1                = 1
PROPGFX_POLYY1                = 2
PROPGFX_POLYX2                = 3
PROPGFX_POLYY2                = 4
PROPGFX_POLYX3                = 5
PROPGFX_POLYY3                = 6
PROPGFX_POLYLN                = 7

'Plot offsets ( in words )
PROPGFX_PLOTC                 = 1
PROPGFX_PLOTX                 = 2
PROPGFX_PLOTY                 = 3

'Circle offsets ( in words )
PROPGFX_CIRCLEC               = 1
PROPGFX_CIRCLEX               = 2
PROPGFX_CIRCLEY               = 3
PROPGFX_CIRCLER               = 4

'Line offsets ( in words )
PROPGFX_LINEC                 = 1
PROPGFX_LINEFROMX             = 2
PROPGFX_LINEFROMY             = 3
PROPGFX_LINETOX               = 4
PROPGFX_LINETOY               = 5


PROPGFX_FILLED_MODE           = 0
PROPGFX_WIREFRAME_MODE        = 1
```

PropGFX Programming Reference Guide

PROPGFX_POLY_STOP = 511 ' this is a stop command for the POLY renderer to STOP processing and go back to waiting for starter instruction.
PROPGFX_POLY_SET_BASE = 510 ' this will read the next WORD and use that as the new screen base for the POLY renderer.
PROPGFX_POLY_CLS = 509 ' cls this will clear the screen with 0
PROPGFX_POLY_SET_WINDOW = 508 '
PROPGFX_POLY_SET_8_PATS = 507
PROPGFX_POLY_CLR_WINDOW = 506
PROPGFX_POLY_PLOT = 505
PROPGFX_POLY_CIRCLE = 504
PROPGFX_POLY_LINE = 503

PROPGFX_SCROLL_STOP = 511
PROPGFX_SCROLL_SET_SCR_BASE = 510
PROPGFX_SCROLL_SET_BLK_BASE = 509
PROPGFX_SCROLL_SET_MAP_BASE = 508
PROPGFX_SCROLL_CLS = 507
PROPGFX_SCROLL_SET_MAP_XY = 506
PROPGFX_SCROLL_SET_MAP_SIZE = 505
PROPGFX_SCROLL_SET_WIN_XYWH = 504
PROPGFX_SCROLL_CLR_WIN = 503
PROPGFX_SCROLL_UPDATE_MAP = 502
PROPGFX_SCROLL_SCROLL_UP = 501
PROPGFX_SCROLL_SCROLL_UR = 500
PROPGFX_SCROLL_SCROLL_RT = 499
PROPGFX_SCROLL_SCROLL_DR = 498
PROPGFX_SCROLL_SCROLL_DN = 497
PROPGFX_SCROLL_SCROLL_DL = 496
PROPGFX_SCROLL_SCROLL_LT = 495
PROPGFX_SCROLL_SCROLL_UL = 494
PROPGFX_SCROLL_UPDATE_TP = 493
PROPGFX_SCROLL_UPDATE_BT = 492
PROPGFX_SCROLL_UPDATE_LT = 491
PROPGFX_SCROLL_UPDATE_RT = 490

PROPGFX_SCROLL_BLOCK_1x1 = 0
PROPGFX_SCROLL_BLOCK_2x2 = 1
PROPGFX_SCROLL_BLOCK_4x4 = 2
PROPGFX_SCROLL_BLOCK_8x8 = 3

PROPGFX_SPRITE_OFF_SCREEN = $01f0

PROPGFX_PARALLAX_SPACETILE = $8000 + $20 << 6

SPRX = 0
SPRY = 1
SPRP = 2
SPRC = 3

PropGFX Programming Reference Guide

This page is intentionally left blank to be a seperator.

PropGFX Programming Reference Guide

PropGFX_Object functions.

PropGFX_Start Starts up the Communications cog.

PropGFX_TestValid Does a test send on the pins, to see if it responds to a clk pulse. Effectively testing to see if **PropGFX** is connected.

PropGFX_SendChar(ptr,i) Sends a 32byte buffer (ptr) to character I (which translates to PROPGFX_CHRSET_BASE + i * 32)

PropGFX_PrintChar(c,x,y) Puts a Character (word) c at screen location (y<<5+x)*2. (assumes your display is set to 32 chars wide, if it's 40 wide, then just change the <<5 to *40.

PrintAt(x,y,c) Prints a string pointed to by c to screen location x,y

PropGFX_SendFont(ptr,targ) sends a 1bit font (ptr) to GFX hub-ram at targ, but converts it to 4bpp, for 16colour charmap modes

PropGFX_SendFontPal sends 16 two colour palettes to the first two entries of each 16 colour palette, (ie colours goto 0,1, 16,17, 32,33, 48,49 etc.)

WaitVSync Waits for Vsync from **PropGFX**

SetComMode(mode) this sets comms mode to (0 = serial mode, 1 = databus mode)

SendBuf(src,dst,len) this sends a buffer (src) of size in bytes (len) to (dst) on **PropGFX**

GetID This gets the 4byte ID of the **PropGFX** (**"L001"** for **PropGFX Lite** and **"V001"** for **PropGFX VGA**)

RequestID this returns the previous GetID's value first byte eg **"L"** or **"V"** to check what you're connected to.

RequestIDVersion this returns the previous GetID's value in full in a Long

Debug(onoff) This turns the **PropGFX**'s debug LED on or off, (1 = on, 0 = off)

WaitSendBuf(src,dst,len) This waits for the **PropGFX** to be idle, then sends a buffer, but doesn't wait for completion before returning.

WaitIdle This waits for the **PropGFX** to be idle.

CogStart(cogpar,cogaddr,cogidx) This restarts a single Cog sending it's cogpar as it's PAR paramter, cogaddr is **PropGFX** hub-ram location to take code from and cogidx is what cog to restart.

PropGFX Programming Reference Guide

SendPropPal(src,dst,len) This sends a Propeller palette to **PropGFX** but automatically converts pal data to **PropGFX VGA** if it's connected to a **PropGFX VGA**

SetPropPal(ptr) this sets up **PropGFX** hub-ram (ptr) to the 256 colour values for Propeller tv output palette.

R_W_EEPROM(rdwr,src,dst,len) this reads/writes to EEPROM (if rdwr = 0 it reads len bytes from EEPROM at src, to hub-ram at dst, if rdwr = 1 it writes len bytes from hub-ram at src, to EEPROM at dst.

GetBuf(src,dst,len) this gets a len byte buffer from **PropGFX** at src, and puts it in dst (on host)

ResetLite(softhard) this resets the **PropGFX** cogs, if softhard = 0 it only resets tv + driver cogs but NOT the comms cog, if softhard = 1 it does a full cold-reset.

DoScroller this signals to the Scroller cog, to go through it's command list, and waits for it to finish.

DoPoly this signals to the Poly cog, to go through it's command list, and waits for it to finish.

FillWords(src,val,len) this fills len words in **PropGFX** hub-ram at src to the value of val.

MoveWords(src,dst,len) this moves len words in **PropGFX** hub-ram from src to dst.

PokeByte(addr,val) this pokes a byte at **PropGFX** hub-ram location addr with value of val.

PokeWord(addr,val) this pokes a word at PropGFX hub-ram location addr with value of val.

PokeLong(addr,val) this pokes a long at PropGFX hub-ram location addr with value of val.

PeekByte(addr) this gets a byte at **PropGFX** hub-ram location addr.

PeekWord(addr) this gets a word at PropGFX hub-ram location addr.

PeekLong(addr) this gets a long at PropGFX hub-ram location addr.

SetPolyWindow(polybase,x,y,w,h) This sets the Poly draw window in the poly cog drawlist polybase, to x,y, with a width of w, and height of h.

SetPolyScrBase(polybase,base) this sets the screen base in the poly cog drawlist polybase to base.

TVTextPrint(x,y,c) This prints a string pointed to by c, to X,Y when using the parallax drivers.

TVTextPrt(c,x,y) This prints a character c to screen location x,y when using the parallax drivers.

PropGFX Programming Reference Guide

SetC64Mode(pal,mode,num3dcogs,filledornot,C64OrNormalSprites) This sets up a C64 style display pal (1=PAL,0=NTSC)
mode =
PROPGFX_C64_CHARMAP_MODE_1BIT = 0
PROPGFX_C64_CHARMAP_MODE_2BIT = 1
PROPGFX_C64_BITMAP_MODE_1BIT = 2
PROPGFX_C64_BITMAP_MODE_2BIT = 3)
num3dcogs =
PROPGFX_0_3D_COGS = 6 ' this is to set the amount of scanline renderers
PROPGFX_1_3D_COG = 5 ' this is to set the amount of scanline renderers
PROPGFX_1_3D_COGS = 5 ' this is to set the amount of scanline renderers
PROPGFX_2_3D_COGS = 4 ' this is to set the amount of scanline renderers
PROPGFX_4_3D_COGS = 2 ' this is to set the amount of scanline renderers
if charmap this turns to
PROPGFX_0_SCROLL_COGS = 6
PROPGFX_1_SCROLL_COG = 5
PROPGFX_1_SCROLL_COGS = 5
filledornot =
PROPGFX_FILLED_MODE = 0
PROPGFX_WIREFRAME_MODE = 1
C64OrNormalSprites =
PROPGFX_NORMAL_SPRITES = 0
PROPGFX_C64_SPRITES = 1

PropGFX Programming Reference Guide

SetAMSMode(pal,mode,num3dcogs,filledornot,C64OrNormalSprites)
display pal (1=PAL,0=NTSC)
mode =
PROPGFX_AMS_BITMAP_MODE_2BIT = 0
PROPGFX_AMS_BITMAP_MODE_4BIT = 1
PROPGFX_AMS_CHARMAP_MODE_2BIT = 2
PROPGFX_AMS_CHARMAP_MODE_4BIT = 3
num3dcogs =
PROPGFX_0_3D_COGS = 6 ' this is to set the amount of scanline renderers
PROPGFX_1_3D_COG = 5 ' this is to set the amount of scanline renderers
PROPGFX_1_3D_COGS = 5 ' this is to set the amount of scanline renderers
PROPGFX_2_3D_COGS = 4 ' this is to set the amount of scanline renderers
PROPGFX_4_3D_COGS = 2 ' this is to set the amount of scanline renderers
if charmap this turns to
PROPGFX_0_SCROLL_COGS = 6
PROPGFX_1_SCROLL_COG = 5
PROPGFX_1_SCROLL_COGS = 5
filledornot =
PROPGFX_FILLED_MODE = 0
PROPGFX_WIREFRAME_MODE = 1
C64OrNormalSprites =
PROPGFX_NORMAL_SPRITES = 0
PROPGFX_C64_SPRITES = 1

PropGFX Programming Reference Guide

SetSpeccyMode(pal,mode,num3dcogs,filledornot,C64OrNormalSprites)
display pal (1=PAL,0=NTSC)
mode=
PROPGFX_SPECCY_MODE = 0
PROPGFX_SPECCY_HICOLOUR = 1
PROPGFX_SPECCY_CHARMAP = 2
num3dcogs =
PROPGFX_0_3D_COGS = 6 ' this is to set the amount of scanline renderers
PROPGFX_1_3D_COG = 5 ' this is to set the amount of scanline renderers
PROPGFX_1_3D_COGS = 5 ' this is to set the amount of scanline renderers
PROPGFX_2_3D_COGS = 4 ' this is to set the amount of scanline renderers
PROPGFX_4_3D_COGS = 2 ' this is to set the amount of scanline renderers
if charmap this turns to
PROPGFX_0_SCROLL_COGS = 6
PROPGFX_1_SCROLL_COG = 5
PROPGFX_1_SCROLL_COGS = 5
filledornot =
PROPGFX_FILLED_MODE = 0
PROPGFX_WIREFRAME_MODE = 1
C64OrNormalSprites =
PROPGFX_NORMAL_SPRITES = 0
PROPGFX_C64_SPRITES = 1

PropGFX Programming Reference Guide

SetBasicMode(pal,mode,num3dcogs,filledornot,C64OrNormalSprites)
display pal (1=PAL,0=NTSC)
mode=
PROPGFX_2BIT_CHAR_MODE = 1
PROPGFX_4BIT_CHAR_MODE = 2
PROPGFX_8BIT_BIG_BITMAP_MODE = 3
PROPGFX_1BIT_BITMAP_MODE = 4
PROPGFX_2BIT_BITMAP_MODE = 5
PROPGFX_4BIT_BITMAP_MODE = 6
PROPGFX_8BIT_BITMAP_MODE = 7
num3dcogs =
PROPGFX_0_3D_COGS = 6 ' this is to set the amount of scanline renderers
PROPGFX_1_3D_COG = 5 ' this is to set the amount of scanline renderers
PROPGFX_1_3D_COGS = 5 ' this is to set the amount of scanline renderers
PROPGFX_2_3D_COGS = 4 ' this is to set the amount of scanline renderers
PROPGFX_4_3D_COGS = 2 ' this is to set the amount of scanline renderers
if charmap this turns to
PROPGFX_0_SCROLL_COGS = 6
PROPGFX_1_SCROLL_COG = 5
PROPGFX_1_SCROLL_COGS = 5
filledornot =
PROPGFX_FILLED_MODE = 0
PROPGFX_WIREFRAME_MODE = 1
C64OrNormalSprites =
PROPGFX_NORMAL_SPRITES = 0
PROPGFX_C64_SPRITES = 1

PropGFX Programming Reference Guide

SetParallaxMode(xt,yt,pal,hx) this sets the display mode to a mode like the Parallax tv.spin driver. xt = number of x_tiles (16 pixels) yt = number of y_tiles (16 pixels)
pal = 1=PAL,0=NTSC, hx = horizontal extent.
Usual setup = SetParallaxMode(16,12,0,8) this gets a 256x192 pixel resolution. 16*6 characters as a character takes up 2 16x16 cells.

Set16ColourSprites this sets 16colour sprite mode, (when using normal sprites mode)

Set4ColourSprites this sets 4colour sprite mode, (when using normal sprites mode)

SetSpriteBase(addr) this sets sprite char base address to addr.

SetBorderColour(colour) this sets the border colour to the Propeller colour value.

SetPALNTSC(mode) this sets display mode to PAL (1) or NTSC (0).