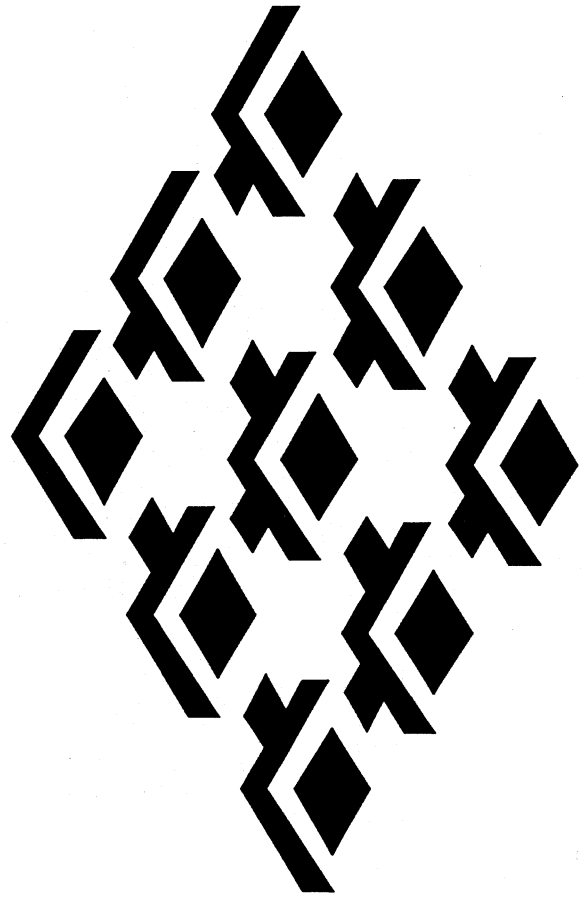
TRANSTECH TTG1

GRAPHICS TRAM
USER MANUAL

VERSION 1.0 4:9:1989



TRANSTECH DEVICES LIMITED
UNIT 17, WYE INDUSTRIAL ESTATE
LONDON ROAD
HIGH WYCOMBE
BUCKINGHAMSHIRE
HP11 1LH
ENGLAND

TELEPHONE (+44) 0494 464303
FAX (+44) 0494 463686

Contents

Chapter 1

Introduction

The TTG1 is a compact medium resolution graphics subsystem TRAM, functionally compatible with the industry standard IMSB007 from INMOS Ltd. The TTG1 fits onto a size 2 TRAM site and offers all the usual links and subsystem control.

The use of the latest surface mount technology, and the Inmos G178 Colour Lookup Table, has kept the component count to a minimum. This level of integration results in increased performance and better reliability.

The TTG1 uses the IMS T800 as a general purpose graphics processor. A total of 1 Mbytes of VRAM is provided for program and data. The four bi-directional serial links on the transputer are ideally suited for downloading data from a distributed image store into the frame store.

All the ram is available for display, but can be used for program and data. The Video Timing Generation performed by the a TMS34061 VSC and is totally user programmable, which ensures the TTG1 can support a wide range of display monitors.

Multiple synchronised framestores may be supported by running the TMS34061 in external sync (slave) mode where one TTG1 is in supplying sync (master) mode.

To maintain compatability with the IMS B007 the TTG1 has two colour modes of operation, six or eight bit DACs. This is software selectable and defaults to 6 bit mode. Six bit mode offers any 256 colours from a pallete of 262144 and 8 bit mode offers any 256 colours from a pallete of 16.7 million.

Each 8 bit pixel defines an offset in the 256 element Colour Lookup Table. Each table entry gives 24 bits of colour, which is used to drive the three DACs.

The TTG1 is in the most part compatible with the Inmos B007 Evaluation board as described in [2].

Chapter 2

Installation

2.1 Package Contents

TTG1
User Guide
Two 360K (MSDOS) Disks
Video cable

2.2 Installing the TTG1

The TRAM conforms to the Inmos TRAM format as defined by Inmos Technical Note [3]. As such the TRAM may be installed on any module mother board designed for Inmos format TRAMs, e.g. Transtech MCP1000-3, TMB08, IMS B008, IMS B014, etc.

The TTG1 is a size 2 TRAM, but only the 16 pins on site 1 are actually used (see block diagram Appendix A). These pins provide the following functions:

Pin	Function
1	Link2Out
2	Link2In
3	VCC
4	Link1Out
5	Link1In
6	LinkSpeedA
7	LinkSpeedB
8	ClockIn(5MHz)
9	Link3In
10	Link3Out
11	GND
12	Link0In
13	Link0Out
14	notError
15	Reset
16	Analyse

Table 2.1: Active Pins

LinkSpeedA is connected to Link0special on the processor, and **LinkSpeedB** is connected to Link123Special.

To determine the parameters necessary to best drive your monitor a simple set-up program is provided. This is a standalone program designed to run from the Inmos afserver. It is configured for a two processor system, in which the root processor is connected to the TTG1 via a single link. Link2 of the root processor must be connected to Link1 of the TTG1. If this configuration is not achievable, refer to section 4.1, which describes the program in more detail.

Connect the three SMB connectors on the supplied cable to the RGB SMB connectors on the TTG1, and the other end to your monitor, see Appendix B Block Diagram for position of R,G, and B. Note that the TTG1 has been designed to work with monitors that have sync on green (Most monitors do this).

You are now ready to install the set-up software, if required.

2.3 Installing Software

It is recommended that a copy is made of both the distribution disks before proceeding any further.

Insert distribution disk 1 into drive A: and run the install batch file. It will ask for the target directory to be defined. This disk contains the TTG Server and should typically be copied to `\TTG1\TTGSERVER` .

The server directory includes a standalone server program and all the occam source. For further details refer to the server guide TTGD001.

Insert distribution disk 2 into drive A: and run the install batch file. It will ask for the target directory to be defined. This disk contains s simple hardware test program (**htest.btl**) and a VTG set-up program (**vtgsetup.btl**). These should normally be installed in the directory `\TTG1` .

2.4 VTG Set-up

For those familiar with the IMS B007 turn straight to Section 3.4 for an explanation of the VTG parameters. For those less familiar and or more anxious to see something on the screen proceed as below.

Apply the necessary power and start the set-up program.

```
afserver -:b vtgsetup.btl [options]
```

For a listing of the options see 4.1.

The following line is displayed on the Host monitor

```
i)ncrement f)rame l)ine p)ixelclock x)size y)size
```

To drive the program, select any of the options from the menu (with a single key press). Whenever p,f,l or n are selected the parameter is incremented/decremented by the current increment value. So to increment the X size for example hit key x repeatedly. To change the increment value first select i)ncrement and and then type either **l** to multiply the incrementor by 10, **s** to divide the incrementor by 10, or **n** to negate the incrementor.

For a full description of the set-up program see section 4.1. The parameters which need to be defined to initiate display, include **frame rate**,**line rate**,**pixelclock**, and **interlace**. These parameters are then used by the program to calculate the final VTG parameters etc.. The basic function of these parameters can be summarised as follows:

line defines the line rate which will typically be between 20 and 40KHz. If you have a multisync monitor then the frame rate can be set to any value within the sync. range of the monitor, otherwise this must be set to the exact sync. value of the monitor.

frame defines the frame rate, which will vary between 45 to 60 Hz. See your monitors specifacations if you are not sure.

pixelclock defines the rate at which pixels are output to the screen. Normally this is fixed by the on board clock module to 25Mhz.

interlace defines the type of scanning that the monitor expects. This is either on or off (Normally off).

If the screen background clears to red then proceed as follows, else there is probably some problem with the RGB leads, or monitor.

1] A white bounding box should appear round the red background.

2] Adjust **line rate** until picture stabilises horizontally.

3] Adjust **frame rate** until picture stabilises completely.

Note: adjusting 2] first saves time as 3] is dependent on 2].

The current parameter settings are displayed on the status line so make a note of these once you are happy with the display. These parameters will be required for any future software development.

For a full listing of the relevant VTG parameters please refer to the TMS34061 Users Guide available from your local Texas distributor.

Chapter 3

Hardware Description

3.1 Overview

The TTG1 supports both the IMS T800 -25 and IMS T800 -30 transputers. The module is configured with 1 Mbytes 100 ns VRAM. The VRAM is used for program and data space as well as the frame store.

The Module can be used with a wide range of display monitors, and is fully programmable through the use of the VSC. The Module can generate 256 colours from a pallete of over 16.7 million.

The display mode is software selectable between 6 and 8 bit DACs.

3.2 Display Resolution

The display resolution is fixed to 512 by 512. The pixel rate is fixed by the use of the on board clock generator and is normally 25Mhz.

3.3 Memory Map

The 1M byte of display memory on the board can be subdivided into four 512 by 512 display areas. Alternatively it can be subdivided into display and program areas as is necessary

Figure 1 shows how the Video and Dynamic RAM is mapped within the address space of the T800, and the control addresses for the Subsystem.

The memory map is illustrated in Figure 3.1

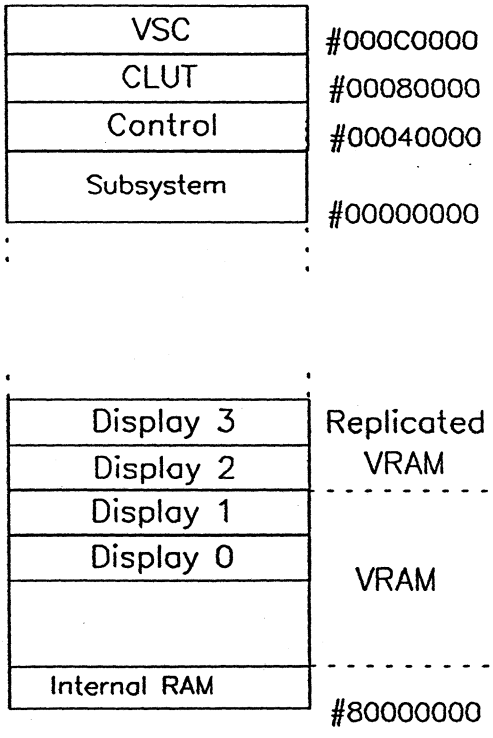


Figure 3.1: TTG1 Memory Map

3.3.1 Screen Addressing

The displays are mapped from address #80080000.

Display 0 at #80080000, 1 at #800C0000, 2 at #80100000 and 3 at #80140000.

The displayed screen is programmable using a two bits of an eight bit register. These bits are mapped onto the least significant bit of the two consecutive display control registers. So if bit 0 of Display Control Reg. 1 is set, and bit 0 of Display Control Reg. 2 is set, then the selected display screen will be 3.

The screen can be addressed as either a two dimensional byte array, or a long single dimensional byte array, which ever is convenient at the time.

The top left of screen (X[0],Y[0]) is at the lowest physical address. This is illustrated in Figure 3.2

The ram wraps around all the way through negative address space, so that if four 512 by 512 displays are necessary they are accessible after the displays 0 and 1.

3.3.2 Control Register

The control register controls six functions, outlined below, all of which are mapped onto bit 0.

1] Subsystem Reset. (Address #00000000)

2] SubSystem Analyse. (Address #00000004)

3] Display Control Reg. 1 (Address #00040000) bit 0 active (l.s.b)

4] Display Control Reg. 2 (Address #00040004) bit 0 active (m.s.b)

5] VSC Reset. (Address #00040008)

6] Colour Mode (Eight/notSix). (Address #0004000C)

3.3.3 Colour Palette

The IMS G178 CLUT (Colour LookUp Table) can be found at address #00080000 and is write only.

The CLUT is mapped onto byte 0 and addressed on word boundaries.

The CLUT has four locations available to the programmer, three of which are usable.

Colour table address register at #00080000. Valid addresses are between 0 and 255.

Colour value register at #00080004, three consecutive writes here will write colour values for the

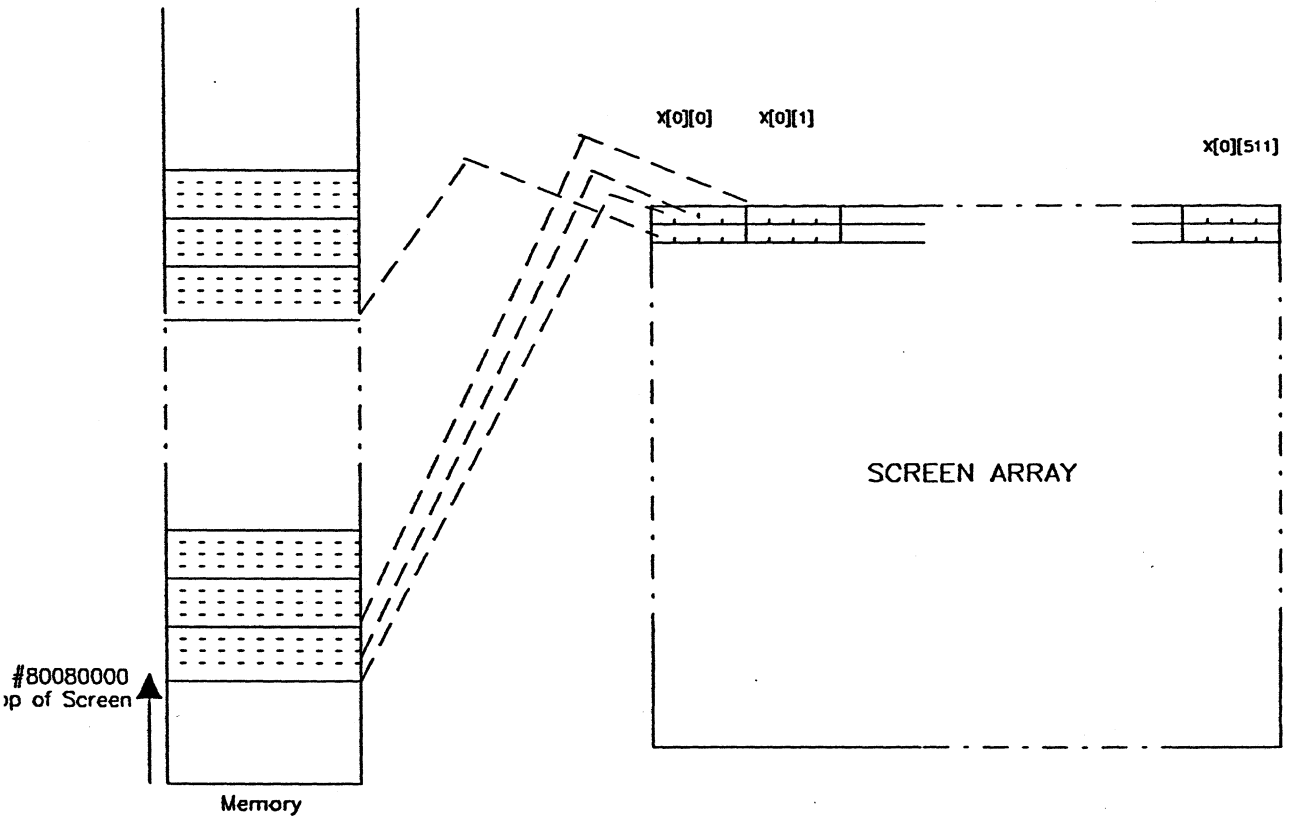


Figure 3.2: TTG1 Screen Addressing

three colour DACs R,G,B respectively at the location specified by the last colour table address write. Note that in six bit mode (default after reset) only bits 0 to 6 are valid.

Colour table mask register at #00080008. Any bit reset to zero in this register will mask (logical AND) any pixel address into the look up table. Writing a zero into this register will make all pixel colours be addressed always through address zero. Writing 255 will allow all addresses available from the pixel data.

3.4 The Video Timing Generator

It is not recommended that the VSC be programmed directly by the user, as a detailed knowledge of the hardware implementation of the TTG1 is necessary to fully grasp the intricacies of this device. The hardware can be hidden from the user by using the supplied software.

The VTG can be configured to generate both interlaced, and non-interlaced display addressing and both internal and external sync can be programmed. Jumper headers are supplied on the TRAM for this application. JP1 pin 1 is the Vertical sync connection to the VSC and JP2 pin 1 the horizontal connection. Pin 2 of both these jumpers is connected to ground.

Normally the user will call the routine initCRTC with the appropriate line and frame parameters and this will be the last the user has to do with this device.

3.5 Event Handling

The Vertical sync signal of the VSC has been used to generate event requests. This enables the use of frame flyback events for synchronising double buffered animated sequences.

A single event is generated at the beginning of frame flyback and it is up to the user to ensure that this event is serviced before the active display period is entered. For a 512 * 512 screen at 50 Hz the user has approximately 2 milliseconds to handle the event.

It should be noted that the transputer event request logic will only be cleared by a process reading the event channel (an event process being rescheduled). It is advisable therefore to handle frame events with a high priority process to ensure that the event is serviced as fast as possible within the non-display window. Care must be taken if other high priority processes may be scheduled at this time.

3.6 Subsystem Control

The TTG1 is provided with a set of subsystem controls. These can be used to control a network of transputers. Three signals are provided: **SubSystemReset**, **SubSystemAnalyse**, and **SubSystemError**.

To maintain software compatibility between TRAMs the subsystem registers start at hardware

Register	Address
SubSystemReset (write only)	#00000000
SubSystemAnalyse (write only)	#00000004
SubSystemError (read only)	#00000000

Table 3.1: Subsystem Registers

address #00000000 , and are mapped as follows:

Setting bit 0 of the relevant register asserts reset/analyse. A '1' read from bit '0' of SubSystemError indicates ERROR.

Note, the TTG1 does not support the logical OR'ing of the Subsystem registers with the TRAMs own reset and analyse inputs. This should not present problems with systems that have a single root host.

Chapter 4

Software Description

4.1 VTG tuning program

The TTG1 is supplied with a simple setup program, designed to ease the task of deriving the optimum VTG parameters for a particular monitor. The source for this program (written in occam) is provided in TDS (D700D) format in (**toplevel.top** on disk 2), and also as a standalone program (**vtgsetup.btl**) which can be run from the Inmos **afserver**.

The standalone program is configured to run on a T414/T800 connected to the TTG1 TRAM. Link 2 of the root processor must be connected to link 1 of the TTG1.

To run the setup program issue the command

```
afserver -:b vtgsetup.btl [options]

valid options   /f <frame_rate>
                /l <line_freq>
                /p <pixel_clock>
                /x <x_size>
                /y <y_size>
```

and the following line is displayed on the Host monitor

```
i)ncrement f)rame l)ine p)ixelclock x)size y)size
```

The optional arguments to vtgsetup define the initial values for the five definable parameters.

To drive the program, select any of the options from the menu (with a single key press). Whenever p,f,l or n are selected the parameter is incremented/decremented by the current increment value. So to increment the X size for example hit key x repeatedly. To change the increment

value first select i)ncrement and and then type either **l** to multiply the incrementor by 10, **s** to devide the incrementor by 10, or **n** to negate the incrementor.

increment select increment to change the current increment/decrement value.

frame defines the frame rate, which will vary between 45 to 60 Hz. See your monitors specifacations if you are not sure.

pixelclock defines the rate at which pixels are output to the screen. Normally this is fixed by the on board clock module to 25Mhz.

line defines the line frequency which will typically be between 40 and 64KHz. If you have a multisync monitor then the line rate can be set to any value within the sync. range of the monitor, otherwise this must be set to the exact sync. value of the monitor.

pixelclock Fixed on the TTG1 to 25 Mhz, defines the rate at which pixels are output to the screen.

4.2 TTG1 Graphics Server

The TTG1 comes complete with a low level graphics server, which supports basic primitives such as DrawCircle, DrawLine, PolyFill, DrawText etc. This Server provides a common interface to the whole range of TRANSTECH graphics devices.

The TRANSTECH Server **TTGS** is documented in document TTGD001.

Library functions of particular importance to the TTG1 are discussed below.

4.2.1 initCRTC

One of the library routines provided in this library is the initCRTC routine. This is a general purpose Cathode Ray Tube Controller setup routine and the interface is compatible with both the medium resolution TTG1 and the high resolution TTG3.

A full listing of the procedure is given in Appendix B (in occam). The procedure interface is defined below:

```
initCRTC(lineFreq,frameRate,pixelClock,scrxSize,scrySize,interlace)
```

The parameters lineFreq,frameRate,pixelClock,scrxSize, scrYsize and interlace, are defined above for the VTG set-up routine vtgsetup(). The values extracted from this set-up procedure should be plugged direct into initCRTC to setup the required display.

Note that on the TTG1 pixelclock, scrxSize and scrySize are all fixed and values supplied to this procedure should be 25000000, 512, and 512 respectively.

Appendix A

Block Diagram

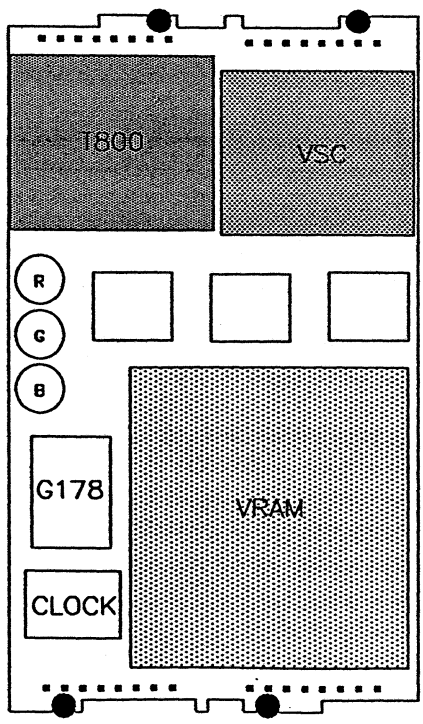


Figure A.1: TTG1 Block Diagram

Appendix B

initCRTC

Listing of initCRTC routine.

```
PROC initCRTC(VAL INT lineFreq,frameRate,pixelClock,x,y,
              VAL BOOL interlace)
  VAL bpw           IS 4 :
  VAL bpwShift      IS 2 :
  VAL mint          IS #80000000 :

  [4] INT controlLatch :
  [(36*4)]INT CRTCReg :          --least sig 8 bits only, 2 word = 1 16 bit reg.

  VAL CRTCRegAddr        IS #000C0000 :
  VAL latchByteAddress   IS #00040000 :
  --VAL latchByteAddress   IS #0 :

  PLACE CRTCReg        AT (CRTCRegAddr    >> bpwShift) -
                          (mint >> bpwShift) :

  PLACE controlLatch   AT (latchByteAddress   >> bpwShift) -
                           (mint >> bpwShift) :
  VAL screenSelect   IS 0 :
  VAL screenSelect1  IS 1 :
  VAL videoReset     IS 2 :
  VAL resetVideo    IS 0 :
  VAL unresetVideo  IS 1 :
  VAL one.second IS 15625:
  VAL tenth.second IS one.second /10:
  VAL hndth.second IS one.second /100:
  VAL thou.second IS one.second /1000: --roughly speaking
  VAL HorizEndSync    IS 0:
  VAL HorizEndBlank   IS 1:
  VAL HorizStartBlank IS 2:
```

```
VAL HorizTotal      IS 3:
VAL VertEndSync     IS 4:
VAL VertEndBlank    IS 5:
VAL VertStartBlank  IS 6:
VAL VertTotal       IS 7:
VAL DisplayUpdate   IS 8:
VAL DisplayStart    IS 9:
VAL VertInterrupt   IS 10:
VAL ControlReg1     IS 11:
VAL ControlReg2     IS 12:
VAL StatusReg       IS 13:
VAL XYOffset        IS 14:
VAL XYAddress       IS 15:
VAL DisplayAddress  IS 16:
VAL VertCount       IS 17:
VAL CR1Hi           IS (ControlReg1 * 8)+4:
VAL CR2Hi           IS (ControlReg2 * 8)+4:
VAL RefreshOff       IS 0:              --0000 0000
VAL HoldAck          IS #00000070:      --0111 0000

VAL ControlReg1data  IS #0400:          --0000 0100 0000 0000
VAL ControlReg2data  IS #7000:          --0111 0000 0000 0000
VAL Blank            IS #2000:          --0010 0000 0000 0000
VAL InterlaceEn      IS #0200:          --0000 0010 0000 0000
VAL ExternalSync     IS #0100:          --0000 0001 0000 0000
VAL EventEnable      IS #0000:          --0000 0100 0000 0000
PROC poke34061 ( VAL INT reg, data )
  SEQ
    CRTCReg[(8*reg)]      := data /\ #FF --least sig 8 bits
    CRTCReg[(8*reg)+4]  := (data>>8) /\ #FF --most sig 8 bits
:
PROC peek34061 ( VAL INT reg, INT data )
  SEQ
    data := CRTCReg[8*reg] /\ #FF --least sig 8 bits
    data := ((CRTCReg[(8*reg)+4]  << 8) /\ #FF00) \/ data
:
PROC externalSync()
  INT regdata:
  SEQ
    peek34061(ControlReg1,regdata)
    regdata := regdata \/ ExternalSync
    poke34061(ControlReg1,regdata)
:
PROC waitFrameFlyback ()
  INT regdata:
  SEQ
    regdata := 0
    WHILE (regdata/\1) <> 1
      peek34061(StatusReg,regdata)
:
```

```
c2 IS CRTCReg [CR2Hi] :
c1 IS CRTCReg [CR1Hi] :
SEQ
  controlLatch [ videoReset] := resetVideo
  VAL timeout IS 20:   --1.28 ms
  INT ch,timenow:
  TIMER clock:
  SEQ
    clock ? timenow
    clock ? AFTER timenow PLUS timeout
  controlLatch [ videoReset] := unresetVideo
  c2 := HoldAck       --do this immediately
  c1 := RefreshOff    --and this
  VAL BackPerCent IS 75:  --50 percent
  VAL Display     IS 512/4:
  INT TotalHorizClocks:
  INT SyncWidth:
  INT BlankWidth:
  INT BackPorch:
  INT FrontPorch:
  SEQ
    TotalHorizClocks := (((pixelClock*10) / lineFreq)+20) /40  --rounded
    SyncWidth  := ((TotalHorizClocks * 75)+500) /1000 --0.075H rounded up
    BlankWidth :=  (TotalHorizClocks - Display) - SyncWidth
    BackPorch  :=  (BlankWidth * BackPerCent)/100
    FrontPorch :=  BlankWidth -  BackPorch

    poke34061 ( HorizEndSync,SyncWidth)
    poke34061 ( HorizEndBlank,SyncWidth+BackPorch)
    poke34061 ( HorizStartBlank,(SyncWidth+BackPorch)+Display)
    poke34061 ( HorizTotal,TotalHorizClocks-1)
  VAL VertSyncWidth IS 3 : -- 3 lines
  INT TotalLines:
  INT VertSync:
  INT VertBlank:
  SEQ
    IF
      interlace
        TotalLines  := ((lineFreq/frameRate) \/ 1)  --odd no lines
      TRUE
        TotalLines  := lineFreq/frameRate  fieldrate
    VertSync    := VertSyncWidth
    VertBlank   := TotalLines - 512

    poke34061(VertEndSync,VertSync)
    poke34061(VertEndBlank,(VertBlank/2)-VertSync)
    poke34061(VertStartBlank,((VertBlank/2)-VertSync)+512)
    poke34061(VertTotal,TotalLines)
  IF
    interlace
      SEQ
```

```
        poke34061(DisplayUpdate,4)
        poke34061(ControlReg1,ControlReg1data \/InterlaceEn)
      TRUE
        SEQ
          poke34061(DisplayUpdate,2)
          poke34061(ControlReg1,ControlReg1data)

    poke34061(ControlReg2,ControlReg2data)
    INT regdata:
    SEQ
      peek34061(VertStartBlank,regdata)
      poke34061(VertInterrupt,regdata)              --start of V blank
    poke34061(DisplayStart,0)               --could use this to scroll
    VAL timeout IS 20:   --1.28 ms
    INT ch,timenow:
    TIMER clock:
    SEQ
      clock ? timenow
      clock ? AFTER timenow PLUS timeout
:
```

Bibliography

[1] "THE GRAPHICS DATABOOK" *INMOS Limited, Inmos Databook Series.*

[2] **Inmos Technical Note** "Design of IMS B007"

[3] **Inmos Technical Note** Tech. Note 25.