Amiga Frame Grabber Documentation

Helge Böhme

COLLABORATORS			
	TITLE : Amiga Frame Grabber Documentation		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Helge Böhme	August 22, 2024	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

Amiga Frame Grabber Documentation

1.1 Amiga Frame Grabber

```
Amiga Frame Grabber

Contents

  The Author
  Disclaimer
 -1 Installation
  0 Introduction - the concept
  1 The Hardware
    1.1 Connectors
    1.2 The power supply
    1.3 The analog part
      1.3.1 Synchronisation
      1.3.2 The sampling frequency
    1.4 The digital part
      1.4.1 The RAMs
  2 The Software
    2.1 Black and white - the simple part
      2.1.1 Image processing
    2.2 Digital filters
    2.3 Colour - the difficult part
  3 The operating manual
    3.1 The options
    3.2 The ARexx port
    3.3 To do
    3.4 Bugs
      3.4.1 Known bugs
  Bibliography
  Thanks
  A History
  B The schematics 1 2
  C The PCBoard layout (Postscript, mirrored) top bottom
  D The place plan
  E The partlist

Features:

· digitizes single video full frames with 768x575 pixels true colour
```

· with scart and cinch connectors for usage with any CVBS source
· works (theoretically) independent from the video system
· signal processing is performed by the Amiga
· preview at 6 seconds per frame on Workbench

The Hardware:

· a little board with 7x16cm^2 (2.75"x6.3") connected to the parallel port
· analog to digital conversion with the TDA8708A at 16MHz
· video frame storage with two video frame DRAMs HM530281R with 324KByte each
· integrated power supply (software controlled)
· schematics and PCB layout available in postscript format

The Software:

· written in AmigaE (by Wouter van Oortmerssen)
· user interface in EasyGUI (by Wouter/Jason R. Hulance):
· preview adjusts automatically to Workbench-Screen's aspect ratio
· many options
· currently supported video standards: PAL
· adaptive three line combfilter
· stores raw data, screen in true colour
· localized
· ARexx port
· AppIcon and AppWindow
· noise-reduction and colour edge enhancement
· supports OCS to AGA, 68k
· runs on AmigaOS 2.1 and above
· successfully tested on:
    - Amiga4000LC040 KS39.106(WB3.0)AGA 2C16F1.7HD
    - Amiga500 KS37.175(WB2.1)OCS 0.5C2.3F0HD

Have a look at my homepage:

http://www.tu-bs.de/~y0001729

1.2 installation

Installation:

· copy "Env/GrabPrefs" to "Env:" (not really necessary)
· copy directory "FGrab" wherever you want

Software, that might not be installed:

· reqtools.library (Nico François, Magnus Holmgren)
· gadgets/tabs.gadget (bypassed, if not present)
· post.library to view and print the schematics and PCB layout
  (HWGPost (Heinz Wrobel) and PostV2 (Christian Eibl) preferred
  - both in aminet:text/print)

Test:

· run "ZonePlateStream" (save the file wherever you want)

· run "FrameGrab" and load the file for testing

1.3 introduction

0 Introduction - the concept

How do I get television (or video) pictures into my Amiga?
This question preoccupied me for a long time. My studies of electrical
engineering helped me along little bit...

Realtime or not or what?

There are three kinds on video digitizers:

Slow scan digitizers: Digitize one or a few pixels per video line
(64$\mathrm{\mu}$s). Therefore a still picture in RGB format must exist. That isn ↩
't actually
realtime, it is comparable to scanning a photography.

Realtime digitizers: Perform the whole signal processing part with
their hardware. The formatting of the data to an Amiga specific format has to
be done also onboard. The whole thing is to be written to chip memory using
DMA. An expensive piece of hardware either for the processor or Zorro bus.

Frame grabbers: These devices can read video signals in realtime and
process them, not realtime. They produce still frames out of running video.

Considering the advantages and disadvantages of all these, the concept for my
frame grabber stands ...

The hardware is very small and simple because of cost and complexity. An
external device is connected to the parallel port of the amiga. Therefore it
can be used with other platforms (if the software is ported).
The tasks of the hardware are restricted to these terms:

· selecting and amplifying the video signal
· synchronisation
· digitising
· data buffering
· data transfer to the amiga

The software must do the processing of the raw video data. The main tasks are
filtering - dividing the brightness signal (the luminance) from the colour
signal (the chrominance) - and chroma decoding (the so called RGB splitting)
using the desired video standard (PAL, NTSC, SECAM, MAC, ...). The advantages
of software signal processing are more flexibility, almost infinite
expandiblity and higher precision of the calculation. Also there is
displaying the picture on the screen, saving it in different formats,
communication with the user and so on.

1.4 hardware

1 The Hardware

If you want to understand everything, that is described here, knowledge about electronics and video signals is recommended. The last thing can be found in the bibliography. Understanding isn't necessary for building the circuit, but knowledge of using a soldering iron (the board is two sided, uses SMDs and has over 100 via's).

Connectors

Image: The Framegrabber outside (the Strapu plastic case).

1.5 power

1.2 The power supply

The power supply is built conventionally: clamp, fuse, transformer, diode bridge, capacitor, voltage regulator (5V), capacitor. And not conventionally: another voltage regulator (5V) and another capacitor. This part is doubled to decouple the analog power supply from the digital. The input capacitor must have a value of at least 2200$\mathrm{\mu}$F, otherwise it hums, that means, ↩ the video signal is crippled every 10 milliseconds.

Another feature is the software controlled power switch affecting the primary side. A opto triac is controlled by the /BUSY line of the parallel port using the 5V pullup and a PNP transistor. The opto triac of type TLP3042 has a zero crossing detector integrated, so there are no voltage bounces during switching.

To protect the triac from over-voltage there are provided two devices: A 300V varistor in parallel to mains shortens bursts from the power grid (e.g. from flashes of lightning). A trasient diode in parallel to the transformer shortens the bursts switching the inductor off.

1.6 analog

1.3 The analog part

The main part is the analog digital converter TDA8708A, it's also the bridge to the digital part. The TDA is a special ADC for video signals with a maximum operating frequency of 32MHz. To make sure, the level is perfect there is a self regulating input amplifier (it can also adjust the offset). It's possible to connect upto three video sources and select one of them by software control. The TDA is connected as shown in the data sheet.

But one after another: The video signal from the scart connector or one of the cinch connectors goes after line adjustment and capacitive coupling to the input pins of the TDA. One of them is selected by two control lines and the dedicated signal goes to the amplifier. There, it's amplified and offset controlled by two capacitors (this means the voltage over them). So the level

to the ADC is always perfectly adjusted (black is black - a level of 64 and white is white - a level of approx. 213). The charge on the capacitors are controlled by the digitized video signal, therefore the TDA has two digital comparators. The principle is simple: If the signal is too dark or too light, the charge on the first capacitor is varied, if the contrast is too high or too low, the charge on the other capacitor is varied.

Now, the video signal passes a simple passive filter to filter out high frequencies to satisfy the sampling theorem (the sampling frequency is 16MHz - why? go further). Before the filter another line connects the signal to the sync separator (LM1881). After the filter the signal is digitized and goes to the input data bus.

1.7 synchronisation

1.3.1 Synchronisation

A never ending data flow is not very useful, if you do not know when, and what data
is there, inorder to synchronise to the video frames, more data is separated from the
analog video signal. The sync separator LM1881 separates the parts of the video signal called blacker than black: the synchronisation pulses. To provide the LM on its CVBS input, the video signal is current amplified and connected with the specified resistance of 75 ohms to a low pass filter. I'm not really satisfied with the LM1881, because it's not possible to separate the sync pulses from noisy signals clearly (there are comparable from the company élantec - EL 1881, EL1882, EL4581, maybe they are better). The LM provides the following digital signals:

CSYNC Composite sync, plain synchronisation pulses (is used together with BURST by the TDA).

VSYNC Vertical sync, marks the beginning of a video field (50Hz), resets the memory to the lowest address.

ODD/EVEN shows, if the actual field is an odd or even, the later selects one of the two memory chips.

BURST marks the position of the chroma burst in the backporch of the horizontal blanking period (actually it's the signal of a monoflop, triggered by the trailing edge of the horizontal sync pulses). BURST is used by the TDA together with CSYNC (both inverted and CSYNC delayed -> two following, not overlapping high pulses). These signals control the amplifier regulation.

1.8 frequency

The sampling frequency

In digital video signal processing there are three possible selections for the sampling frequency:

Clock locked on line frequency: every video line (beginning with the horizontal blanking period) is divided in a constant count of pixels. It's using 15625Hz line frequency and 13,5MHz samping frequency exactly 864 pixels. A phase locked loop (PLL) is necessary to synchronize the sampling frequency to the line frequency.

Clock locked on chroma freqency: is's suggestive, to sample at four times the chroma frequency of about 4.43MHz, because chroma demodulation will be very simple (another PLL is necessary). Another disadvantage is less adaptability to several colour television standards (the chroma frequency is 3.58MHz at NTSC).

Free running clock: the sampling frequency is totally unlocked from the video signal. Nevertheless you should choose an integer ratio to the line, to make sure the field dots are almost orthogonal.

To satisfy expense and practice I have chosen the last one at a sampling frequency of 16MHz, there are exactly 1024 samples per line (a whole video frame fits in 625KBytes). As this clock is not exactly synchronised to the video signal, it may happen that the picture appears slighly slanted on the screen.

1.9 digital

1.4 The digital part

The whole circuit is controlled by the parallel port (there are no switches or buttons to press). First of all the SEL line is used to choose the data direction controlled by the Amiga (SEL="1": Amiga -> FrameGrabber, SEL="0": the other way). The data is always received by the input registers (IC9, IC10), transmitted directly from the RAMs. The standard handshake protocol of the parallel port is used, so data is delivered and demanded at a negative going /STROBE pulse. The /ACK pulse is driven directly from /STROBE just inverting it - the circuit is fast enough to receive/transmit the data. I want to hint at this place that it's necessary to make a line adjustment for /STROBE and not connect /ACK directly to the RCK inputs of the RAMs. In most cases the FrameGrabber is connected with a cable to the Amiga port, there are signal delays and reflections, the RAMs trigger several times at one pulse and data is lost.

What's stored in the input registers exactly:

bit	name	function
D0	ReadReset	resets the read address of the RAMs
D1	ReadOdd/Even	chooses the RAM to read from
D2	I0	
D3	I1	both choose the input channel of the TDA8708A
D4	/Freeze	signals whether new data is to be stored in the RAMs or not

The signal /Freeze is buffered by another flipflop, it's triggered by the ODD/EVEN signal. This prevents freezing the picture part way through a frame.

Finally there is a simple reset generator (using a R-C combination and an

inverter), it resets the flipflops and the RAMs after switching the circuit on.

1.10 rams

1.4.1 The RAMs

The HM530281Rs are 331776-word x 8-bit frame DRAMs. They are especially designed for digital video applications. They have two completely independent data ports (one for writing and one for reading), two internal address counters (there is no address bus) and an integrated refresh controller. You don't have to look after nothing else but the data going in and out, the pertinent clock pulses and the reset pulses for the internal counters. That sounds good and it works perfectly.

The disadvantage of these chips is: they are less available, the usual distributors for electronic components selling to private persons don't have them (in germany). You should try Hitachi industry distributors found on the european Hitachi website:

http://www.hitachi-eu.com/hel/ecg/dist.htm

Image: The framegrabber, opened (the changes are integrated in the layout yet).

1.11 software

2 The software

I'll reduce the description of the software to the signal processing parts.

2.1 Black and white - the simple part

There are two modes of displaying the data:
1. the raw output of data (it's displayed just like it comes from the hardware - maybe it will be dithered to the available grey values),
2. cropped to the usually visible part of the image, scaled to 4:3 aspect, black level clamped, contrast regulated and maybe filtered output.

The raw type of output also displays the usually invisible parts of the video frame. There are:

- the synchronisation pulses (the value "blacker than black") including the so called equalizing and serration pulses of the vertical blanking interval
- the front- and backporch of the horizontal blanking period - they are usually black, but now dark grey to distinguish them from the sync pulses
- the chroma burst in the backporch
- additional signals in the vertical blanking period (teletext, VPS and some measuring signals).

Image: The blanking period

1.12 imageprocess

2.1.1 Image processing

First of all the video frame is composed of two interlaced fields, they are read and processed one after another:

Centreing or synchronisation: the image clip is positioned at the appropriate place, the 896x575 pixels of the image are cutted out of the 1024x625 pixels.

Filtering: If it's a colour frame, the colour carrier can be removed by digital filters (have a look at next chapter for that).

Scaling: The image is shrunk to 4:3 (768x575 pixels).

Black level clamping: The black level is extracted from the backporch (usually 64), this value is subtracted from the visible pixels.

Contrast regulation: The black level corresponds 300mV (above the sync pulses) of the video signal. In this case white corresponds exactly 1V, the digital representation of white is calculated from this ratio (usually 213). The pixel values are calculated to the total range from 0 to 255.

Noise reduction: A noisy picture could be smoothed by an IIR (infinite impulse response) low pass filter. To blur the image not too much, the filtering is turned off, if image edges are detected. (This method is much more effective and easy to implement than a median filter).

1.13 filter

2.2 Digital filters

There are two ways to remove the chroma carrier (4433643.75Hz at PAL - this corresponds to a period of ~3.6 pixels) from the video signal:

1. one dimensional filtering - the usual way in simple TV sets
2. two dimensional filtering - the way could only be achieved using digital signal processing with realistic complexity

One dimensional filtering in digital video is usually processed using a FIR (finite impulse response) filter. A few neighboured pixels of a video line are each multiplied with a factor and then added together. The quantity of the pixels used sets the order of the filter.

An example: A lowpass filter is realised using for all n pixels the same factor 1/n (this equals averaging the pixels).

Two dimensional filters are also called combfilters. The name combfilter results from the comb like one dimensional spectrum, it fits exactly the spectrum of the chroma carrier. A two dimensional filter considers also the neighboured lines (the chroma carrier in the second line before and after is shifted by 180\textdegree{}). The lines have to be buffered for reference, this explains
the necessity of digital signal processing. If there are colour of brightness

edge from one line to another they are averaged (hanging dots), detecting these and cause the desired reaction is the exercise of so called adaptive comb filters.

Image: The test pattern, two dimensionally filtered

The test pattern (the ".video" file was generated by the program "ZonePlateStream(.e)") has above the usual greyscale and colour bars the so called zone plate. The zone plate contains all frequencies which are possible within a video signal, the frequencies are arranged so that the 2 dimensional fourier-transformation of it is again a zone plate. So the zone plate shows what's happening in the frequency domain also in the spacial domain. The image shows which frequencies are removed by the comb filter. The advantage is keeping the vertical structures which results in a much sharper picture.

Finally the black and white image (the luminance) is the difference of the raw video signal and the colour signal (the chrominance).

1.14 colour

2.3 Colour - the difficult part

First of all the quality of colour processing depends much from the quality of the filters. Is the filter not perfect, luminance and chrominance are not divided correctly and there are cross effects. Far from that the chroma decoding contains these topics:

Phase locking: The chroma signal is quadrate amplitude modulated,
 if contains two components (they are called U and V if PAL) which are
 shifted by 90\textdegree{}. For demodulation is the knowledge of the phase (0\ ↩
    textdegree{}) necessary,
 therefore a reference signal takes place in the backporch of the horizontal
 blanking period (the chroma burst - about 10 oscillations at 135\textdegree{}) to
 synchronize the (digital) oscillator. In practice the burst is demodulated
 the usual way the then the divergence is compensated.

Synchrondemodulation: The colour components U and V are calculated
 from the chrominance C this way:

$$U = s \cdot C \cdot \cos \beta$$
$$V = s \cdot C \cdot (+/-) \sin \beta$$

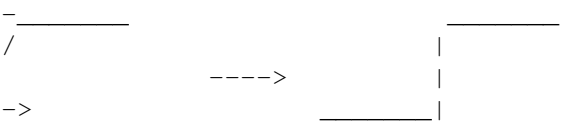
 s is a factor containing the saturation, it's calculated from the amplitude of
 the chroma burst which is saturated by 75%. Its amplitude is usually about 20%
 of the value of white (to black) resulting to approx. 140mV. Switching the
 phase of V form line to line is quality of the PAL system and it's explained
 below. beta is the value of the digital oscillator increment. In practice the
 harmonic calculations are performed using a sinus table with 4096 entries, the
 increment is in this case exactly 1135 (if the sampling frequency is 16MHz).

Low pass filtering of the components: U and V are overlaid by a sin^2 chroma
 carrier noise, it's removed using a FIR lowpass filter.

PAL compensation: To compensate principle phase divergences (and

resulting false colours), the phase of the V component is shifted by 180\ ↩
textdegree{} from
line to line. If the V component is reshifted during demodulation (or just inverted every second line) and the U and V components are averaged from line to line, these phase divergences are compensated totally. This results in a halve chroma resolution in vertical direction, but you can set a threshold for switching this process off for high chroma contrasts.

Colour edge enhancement (optional): The chroma signal has a lower bandwidth than the luminance, that's why the colour edges are averaged. These edges can be detected and increased just shifting neighboured pixels. But if you choose the grade of increments too high, also colour edges will be increased which wasn't sharp.



Dematrixing: R,G,B are calculated from the luminance Y and the components U, V as shown:

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1.14 \\ 1 & -0.37 & -0.57 \\ 1 & 2.04 & 0 \end{pmatrix} \cdot \begin{pmatrix} Y \\ U \\ V \end{pmatrix}$$

1.15 manual

3 The operating manual

Starting the program "FrameGrab", should switch on the hardware. The program will not allow you to interact with the hardware, unless its been detected, but you could still open files and use the program. If another "FrameGrab" already running there are no hardware accesses and no ARexx port for the second one. The program can be launched from Workbench or at the command line interface:

"FrameGrab FILE,ICONIFIED/S,NOMESSAGES/S,INTERRUPT/S,PUBSCREEN/K"

FILE: Name of a ".video" file to load
ICONIFIED: Starts the program as an AppIcon.
NOMESSAGES: Suppresses the most of the warning requesters.
The last two parameters makes the program launching from within an ARexx script simpler.
INTERRUPT: There will be an interrupt routine installed for parallel port accesses. This is slower, but maybe it works better.
PUBSCREEN: The name of a public screen for the GUI (usually it's the default PubScreen).

The program presents itself using the GadTools based EasyGUI (by Wouter van Oortmerssen and Jason R. Hulance):

Image: The FrameGrab main window

Within the main window there is a preview area showing the running video with less quality (one refresh on my A4000LC040 takes approx. 6 seconds)

and some function buttons. These are almost all sticky and show also the
state of the program. The actions of the buttons are explained here:

Open(o) opens (or closes) a ".video" file to display and image processing.
A opened file can be handled exactly like an image directly from the frame
grabber (excluding the synchronisation and freezing).

Save(s) saves the current frame (doesn't care if it's from the grabber or
from a file). The are three possibilities:
1. The raw video stream: the frame will be stored as if it is file
   (625KByte), it could be loaded again with "Open".
2. Render-Screen: The screen containing the rendered image will be stored to
   a IFF-ILBM file as it is (including HAM artefacts, dithering, etc.).
3. True colour: all image processing steps will be performed and the resulting
   24 bit data will be stored as an IFF-ILBM24 file. To buffer one calculated
   field a little bit additional fast RAM is allocated (max 961536 Bytes).

Options(p): The options window will be opened (or closed).

Freeze(f): If a hardware is connected, the running video can be freezed.
This button activates itself during saving and rendering.

Render(r) starts all (selected) image processing steps and displays the
image on the best available screenmode.

Quit(q): No Comment.

There is also a little menu:

About: A little information requester opens.

Iconify: The window closes and an AppIcon appears. You can load ".video"
files just dropping the desired icons on this application icon (or on the main
window) - but this procedure should be already known.

Quit: look above

1.16 options

3.1 The options

The options window is divided in six groups, but the lower row of buttons
exists always:

Save: The options are stored in "ENVARC:GrabPrefs" lasting, used and the
options window is closed.

Use: The options are stored in "ENV:GrabPrefs" temporally, used and the
options window is closed.

Undo: All changes of the options since its opening are made undone.

Cancel: Like undo, the window is closed additionally (the same does the close
gadget in the window frame).

Image: The General options

Get Screenmode: A ReqTools screenmode requester opens, you can choose the screenmode for the render screen, it will be shown by name after selection.

Use HAM: The render screen is of type hold-and-modify if necessary.

Use FS-Dither: Floyd-Steinberg dithering is used for rendering.

Snapshot Windows Now: Stores the positions of all windows.

Image: The Input options

Input Channel: Selects one of three video input ports (Scart, Cinch1 or Cinch2).

Buffer Files: Opened files are stored in RAM, this increases the calculation speed but uses 625KByte additional fast RAM.

Software Synchronisation: FrameGrab tries to synchronize not perfect aligned fields (caused in noisy video signals) line by line (does not really work perfectly).

Image: The Output options

Save as selects the save mode (raw video frame, render screen, true colour or ask).

Fileextensions activates automatically appending of the desired extension (".video", ".ilbm", ".ilbm24") to the filenames (this could still be changed in the filerequester).

File numbering activates automatically numbering of the files.

Image: The Preview options

Floyd Steinberg Dither for the preview window,

Smoothing for the preview window, activates the low pass filter,

Field selects the odd or even field for the preview.

Image: The Luminance options

show Synchronisation Pulses activates displaying and saving of the synchronisation pulses together with the image, deactivates the chroma filter in black and white mode.

Chroma Filter switches the chroma carrier filtering for the luminance on.

Combfilter activates the combfilter (0 is off), selects the adaptability level (a good value is 4).

Noise Reduction activates and selects the intensity of the noise reduction (adaptive IIR low pass filter). This has also affect to the chrominance).

Picture Level selects the level of the video signal (this depends on

the video source - usually it's 700mV from black). It does the same as a contrast
 controller.

Calculates the best picture level of the current image.

Image: The Chrominance options

Chromadecoder selects the chromadecoder. Currently there are only "none"
 (black and white) and PAL (as described in ch. 2.3.

PAL-Compensation activates the line by line averaging of the chroma components
 and selects the threshold (good values are from 4 to 6).

Chroma-Level selects the well known saturation (good values are from 40 to
 50).

Colour Edge Enhancement activates itself. Values larger than two are used
 carefully because also unsharp colour edges are enhanced.

1.17 The ARexx port

3.2 The ARexx port

After starting the program an ARexx port called "FGRAB" is installed. The
following commands are supported ([optional] <required>):

FREEZE switches between running and frozen video. This makes only sense if
 a hardware is connected and no file loaded.

RENDER The current video frame is rendered.

QUIT ...

ICON (De-)iconifies.

SAVE [file] Saves the picture at the current mode in the last (auto-numbered)
 or given file.

OPEN [file] Opens the given (or last) ".video" file

CLOSE The opened file is closed and the parallel port is used again (if the
 hardware is present).

CALCPICLEVEL Recalculates the picture level (does the same as the button in
 the luminance options window).

SET <var> <value> Sets the given variable to the desired value.

QUERY <var> Reads the given variable and assignes the value to the ARexx
 'RESULT' variable.

3.2.1 The variables

Variable	Values	Comment

```
MODEID       | integer                    | the screenmode identifier
HAM          | TRUE|FALSE                 |
FLOYD        | TRUE|FALSE                 |
INPCHANNEL   | SCART|CINCH1|CINCH2        |
FILEBUF      | TRUE|FALSE                 |
SOFTSYNC     | TRUE|FALSE                 |
SAVEAS       | ASK|VIDEO|SCREEN|TRUECOLOR |
FILEEXT      | TRUE|FALSE                 |
FILENUM      | TRUE|FALSE                 |
PREVIEWFS    | TRUE|FALSE                 |
PREVIEWSM    | TRUE|FALSE                 |
PREVIEWFR    | EVEN|ODD                   |
SYNCS        | TRUE|FALSE                 |
CHRFILTER    | TRUE|FALSE                 |
COMBFILTER   | integer[0-10]              |
NOISERED     | integer[0-4]               |
PICLEVEL     | integer[50-90]             | equals. 500mV to 900mV
CHROMADEC    | NONE|PAL                   |
PALCOMP      | integer[0-10]              |
CHROMALEVEL  | integer[0-100]             |
COLEDGEENH   | integer[0-4]               |
BUSY         | TRUE|FALSE                 | READONLY preview is still drawing
FROZEN       | TRUE|FALSE                 | READONLY picture is frozen
ICONIFIED    | TRUE|FALSE                 | READONLY
DISPMESSAGES | TRUE|FALSE                 | show all warning requesters
FILENAME     | string                     |

A example script
An image sequence is stored.
```

1.18 todo

```
 3.3 What's to do?

This is the not complete list of future features of the frame grabber:

· bugfixes
· improved chroma filters (maybe in frequency domain)
· optimise the HAM mode
· better synchronisation
· graduation rectification (gamma correcting)
· measurement: oscillo- and vectorscope, spectrum analyser
· reading out of additional information in the blanking period (teletext, VPS,
  line 23)
· PALplus (including 16:9 screen)
· NTSC, SECAM, MAC, ...
· descrambler
· speed optimization
· ...
```

1.19 bugs

3.4 Bugs

Bugreports are best sent to me by EMail: h.boehme@tu-bs.de

If I should include other video standards (NTSC, SECAM, ...) in the software, you have to send me packed ".video" files containing captured frames of the desired standard. If I should include a graphicscard display, feel free to send me a graphicscard ;-) (maybe its technical documentation at least).

3.4.1 Known bugs

· The SEL line of the parallel port is connected within the Amiga to RI of the serial port. Maybe there are problems with modems (very unlikely) -> disconnect the frame grabber if the modem doesn't work.

1.20 bibliography

Bibliography

Sorry, it's mostly german:

· Fernsehtechnik und Bildübertragung (Teil I & II), Prof. Dr.-Ing. U. Reimers, Vorlesungsscripte, Institut für Nachrichtentechnik, Technische Universität Braunschweig
· Digitale Signalverarbeitung in Fernsehgeräten, Paschko, Verlag Technik
· Von PAL zu PALplus, Dobler, Verlag Technik
· Digitale Filter in der Videotechnik, Schönfelder (Hrsg.), Drei R
· Computer-Sehen, Prof. Dr. Friedrich M. Wahl, Vorlesungsscript, Institut für Robotik und Prozeßinformatik, Technische Universität Braunschweig
· Digitale Bildsignalverarbeitung, Prof. Dr. Friedrich M. Wahl, Springer
· Data sheet TDA8708A, Phillips IC02, http://www.semiconductors.philips.com/acrobat/2031.pdf
· Data sheet HM530281R, Hitachi, http://www.hitachi-eu.com/hel/ecg/pdf/01_063.pdf
· Data sheet LM1881, National http://www.national.com/search/search.cgi/design?keywords=lm1881
· High-Speed CMOS Data, Motorola

1.21 thanks

Thanks

AmigaE & EasyGUI: Wouter van Oortmerssen & Jason R. Hulance

Betatesters: Björn Haake, Lars Unger

Proofreader of the english docs: Chris Thornley

1.22 history

Version History

Hardware

Revision 1.0

First working version

Revision 1.1

BUG The signal after the low pass filter must not be connected to SCART
directly -> added an amplifier and line adjustment!

BUG Bad reflections within the parallel port cable of just 1.5m -> added
a line closing to Strobe (R-C low pass). Additionally the read clocks of
the RAMs must not be connected to acknowledge of the port (back-reflected
waves retrigger them) -> doubled the circuit producing this signal!

BUG The LM1881 syncseparator has no good results if the video signal is
noisy -> added low pass filter.

BUG Very sad - the TDA8708A input ports GateA and GateB are active high,
CSync and Burst from the LM1881 are active low, that's why the TDA stayed
in controlmode 1 and loss of image quality -> added inverters!
The GateB signal should be a little bit delayed (about 1 microsecond)!

NEW Added a PNP transistor circuit to BUSY -> now software controlled on/off
switching in possible.

Software

Version 1.0

First release

Version 1.1

NEW Localisation

NEW AppWindow and AppIcon

NEW Command line and menu

NEW ARexx port

Version 1.2

IMPROVED Parallel port: replaced the interrupt routine by a burst routine
-> twice as fast preview!

NEW One dimensional noise reduction

NEW Colour edge enhancement

BUG Preview smoothing didn't work correctly

IMPROVED Combfilter: works much better at vertical colour edges now.