



AN1183 APPLICATION NOTE

AUTO-SIZING & AUTO-CENTERING Using TMU Cell from ST72774 Family

by Pascal JANIN & Hervé PIERROT - CMG IDD Monitor Application Lab

This Application Note describes how to use the **TMU** (Timing Measurement Unit) cell which is implemented in the ST72774 family of Monitor-dedicated Microcontrollers. The TMU cell implementation is reviewed in details on a real monitor chassis, through 2 new features called **Auto-Sizing** and **Auto-Centering**.

Note : Along the entire document, **T1_{AV}** and **T2_{AV}** are the values returned by the TMU cell.

Table of Contents

1 WHY AUTOMATIC SETTINGS	5
1.1 User settings	5
1.2 Re-adjustments between screen modes	5
1.3 Automatic adjustments	6
2 AUTO-ADJUSTMENT BASICS	7
2.1 Horizontal centering	7
2.2 Horizontal sizing	8
2.3 Vertical centering and sizing	9
3 THE NEW TMU CELL	10
3.1 Principle	10
3.2 Active video signal	10
3.3 Start a measurement	11
3.4 Limitations and constraints	12
3.4.1 Overflow	12
3.4.2 The Active Video signal is too early	12
3.4.3 The Active Video signal is too late	12
3.4.4 Unstable picture	13
3.4.5 Measurement-resolution limitation	14
4 HARDWARE AND SOFTWARE IMPLEMENTATION IN A REAL MONITOR	15
4.1 H/W and S/W prerequisites	15
4.2 Overall S/W behaviour	16
4.3 Horizontal autocentering (HPOS)	17
4.3.1 Successive Approximations Algorithm	18
4.3.1.1 Computation time : How to keep it minimum	18
4.3.1.2 Measurements of T1AV/T2AV and preliminary tests	18
4.3.1.3 UPDOWN flag setting	18
4.3.1.4 HPOS setting update	19

4.3.1.5	Avoid “never-ending” loops	19
4.3.1.6	Complete HPOS algorithm	19
4.3.2	Measurements & Computation Algorithm	22
4.3.2.1	First measurement : rough centering	22
4.3.2.2	Second measurement : left shift	22
4.3.2.3	Third measurement : right shift	22
4.3.2.4	Optimum centering computation	22
4.4	Horizontal autosizing (HSIZE)	23
4.4.1	Horizontal deflection formulas	24
4.4.2	Amplitude formulas	25
4.4.3	Operating mode	25
4.4.4	Optimum HSIZE setting formula	26
4.4.5	Measurement of Td' and T'	27
4.4.6	Additional compensation parameter	28
4.4.7	Final HSIZE formula	29
4.4.8	Software implementation	29
4.5	Vertical autocentering/autosizing (VPOS, VSIZE)	31
4.5.1	Vertical sawtooth formula	31
4.5.2	VSIZE setting	32
4.5.3	VPOS setting	33
4.5.4	Complete vertical formula	33
4.5.5	Reference timing	33
4.5.6	Operating mode	36
4.5.7	Integer formulas	36
4.5.8	Software implementation	38
5	SET-UP & MEASUREMENTS INSIDE THE MONITOR	39
5.1	Horizontal centering (HPOS)	39
5.1.1	The picture is shifted too far to the left	39
5.1.2	The picture is shifted too far to the right	39

5.1.3	The picture is perfectly centered	40
5.2	Horizontal sizing (HSIZE)	40
5.2.1	Measurement presets	41
5.2.2	Current-amplitude measurements	41
5.2.3	Amplitude-measurement of AVopti	41
5.2.4	Measurements of HAMPMIN and HAMPMAX	42
5.3	Vertical centering/sizing (VPOS/VSIZE)	44
6	END-USER INTERFACE : FINAL TIPS AND FINE-TUNING	45
6.1	Pictures that work best	45
6.2	End-user fine-tuning	45
6.2.1	Optimum HAMP size	47
6.2.2	Optimum VPOS and VSIZE	47
7	ANNEX	48

1 WHY AUTOMATIC SETTINGS

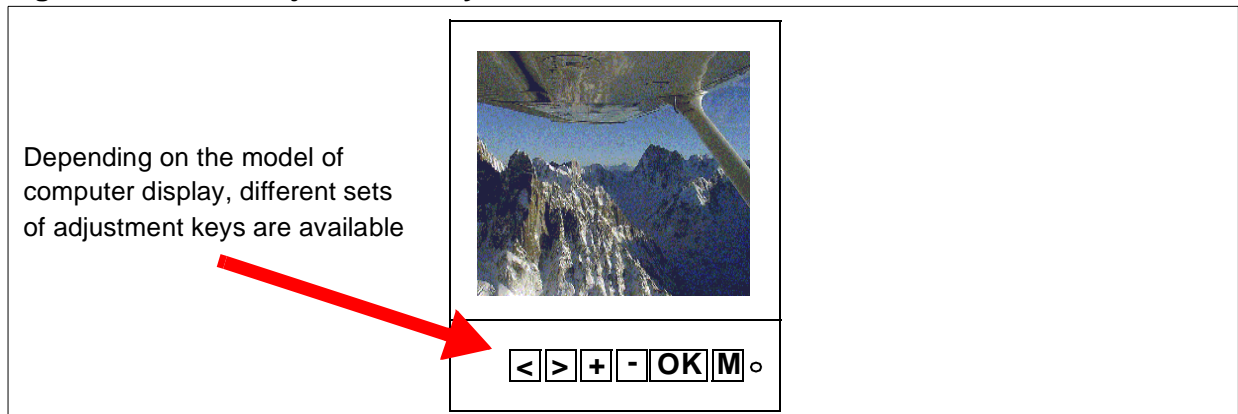
1.1 User settings

Every computer display provides adjustments called *user settings* allowing the users to adjust the picture being displayed according to their taste. If these user settings cover the usual brightness and contrast, they also also allow other modifications such as:

- n picture size & position (both horizontally and vertically)
- n screen geometry (edges, corners, tilt..)
- n colour temperature
- n OSD language, etc..

For that purpose, some keys (and/or rotary knobs) are provided on the front of the screen :

Figure 1. Manual Adjustment Keys



The user needs to go through several menus and submenus (commonly displayed on screen) to adjust all the needed parameters. This adjustment procedure is fast and convenient, as there are very few keys available with multiple functions assigned per key.

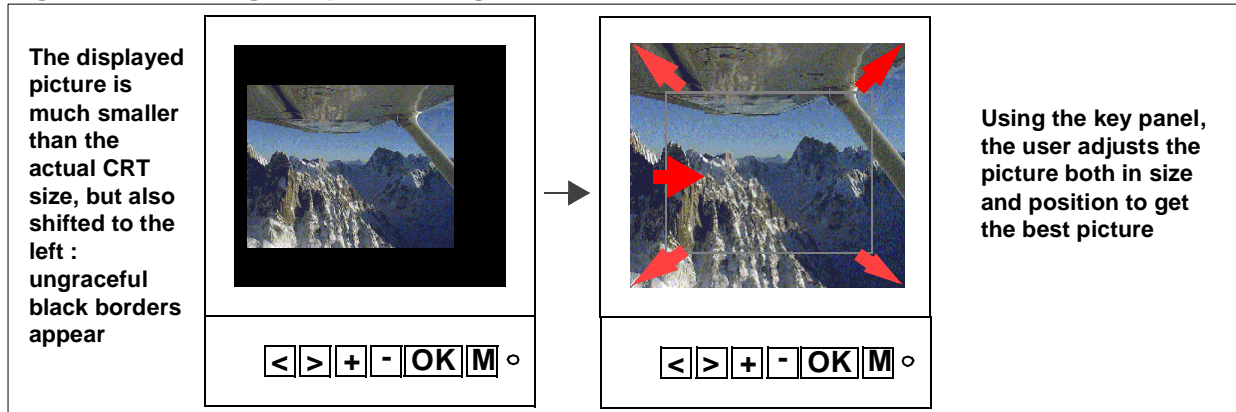
1.2 Re-adjustments between screen modes

Among all, size and position adjustments are often required when switching between video modes: some softwares (mostly games) swap video modes, for example from classic VGA 640x480 pixels to higher resolution 1280x1024 pixels, allowing the display of sharper and larger graphics with more colours.

When changing from one video mode to another, the microcontroller inside the computer display analyzes the new incoming video synchro signals HSYNC and VSYNC, computes their frequencies, and then proceeds to all the necessary internal adjustments. The new picture is displayed in a proper way.

However, the picture itself is rarely (even never) fine-tuned to fit best the real CRT size. The final result is a picture framed in black : large black borders “surround” the real picture, which is therefore shrunk in size and/or not centered. Those black borders represent a waste of display area : they do not display any picture but black.

Figure 2. Resizing / Repositioning the Picture



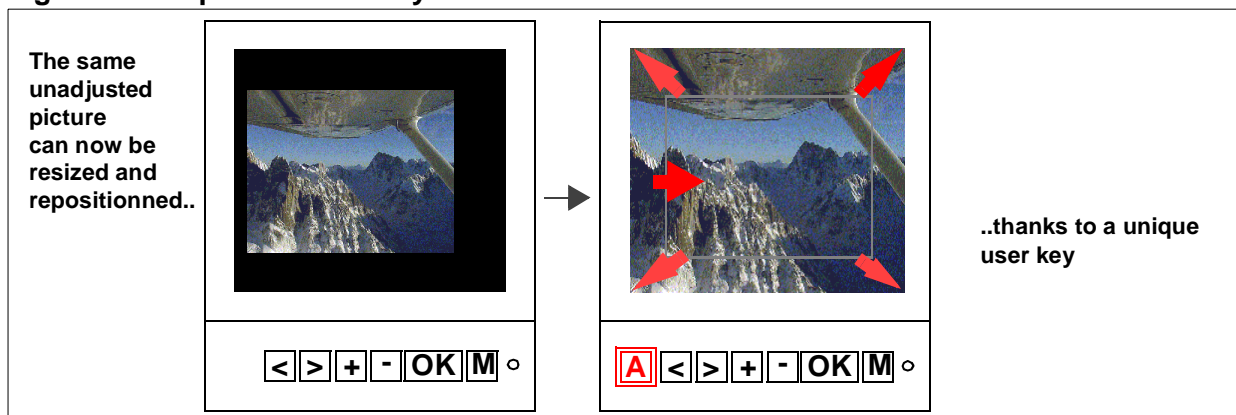
The user then has to re-adjust manually the screen size & position to minimize those borders and get the best sized and centered picture on screen. As this re-adjustment is done by using the control keys, it is most laborious for the user.

Even though most of the computer displays register the video modes already sensed, each new mode requires re-adjustments. Furthermore, certain modes already sensed may not always fit in with *all* softwares using this mode, making repeated re-adjustments necessary.

1.3 Automatic adjustments

With the **TMU cell**, the displayed picture can now be analyzed. Adjustments in size and position, horizontally and vertically are available through the use of a unique user key, at each video mode change. This makes adjustments easier for the user.

Figure 3. Unique “AUTO” Key



2 AUTO-ADJUSTMENT BASICS

A total of 4 different parameters need to be adjusted to obtain the best possible displayed picture on screen :

- horizontal position
- horizontal size
- vertical position
- vertical size

Each of them requires specific handling and is treated separately.

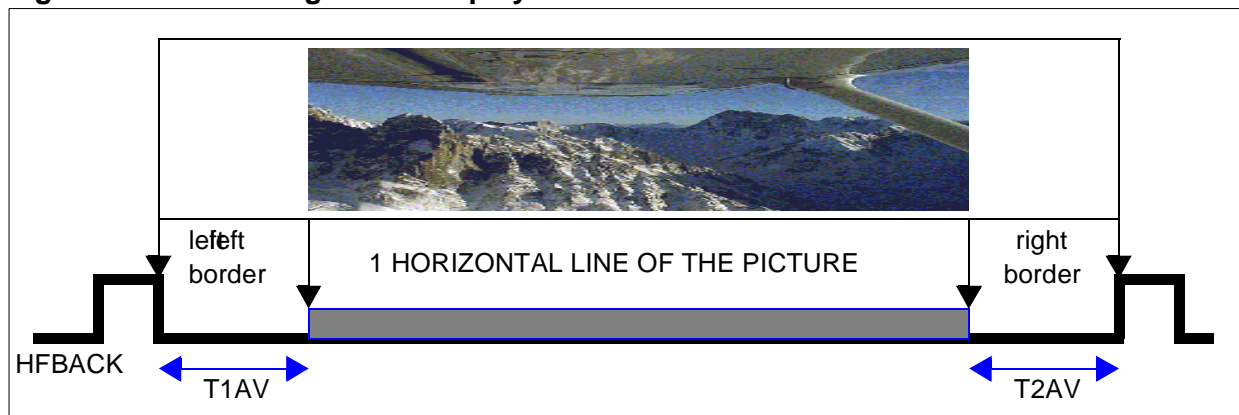
2.1 Horizontal centering

To determine whether a given picture is well centered on screen, the time is measured between :

- the beginning of an horizontal scan line and the left border of the REAL picture ($T1_{AV}$)
- the right border of the REAL picture and the end of an horizontal scan line ($T2_{AV}$)

The beginning and the end of a scan line are delimited by the Horizontal Flyback pulse, **HFBACK**. These *positive* pulses are issued to send the raster back to the beginning of the next scan line. Therefore the *falling* edge of the pulse coincides with the *beginning* of the scan line, while its *rising* edge coincides with the *end* of the scan line.

Figure 4. HFBACK Signal VS Displayed Picture



A quick comparison between $T1$ and $T2$ indicates how the picture is displayed :

- $T1_{AV} < T2_{AV}$: the picture is shifted too far to the left and needs to be moved rightwards
- $T1_{AV} = T2_{AV}$: the picture is perfectly centered
- $T1_{AV} > T2_{AV}$: the picture is shifted too far to the right and needs to be moved leftwards

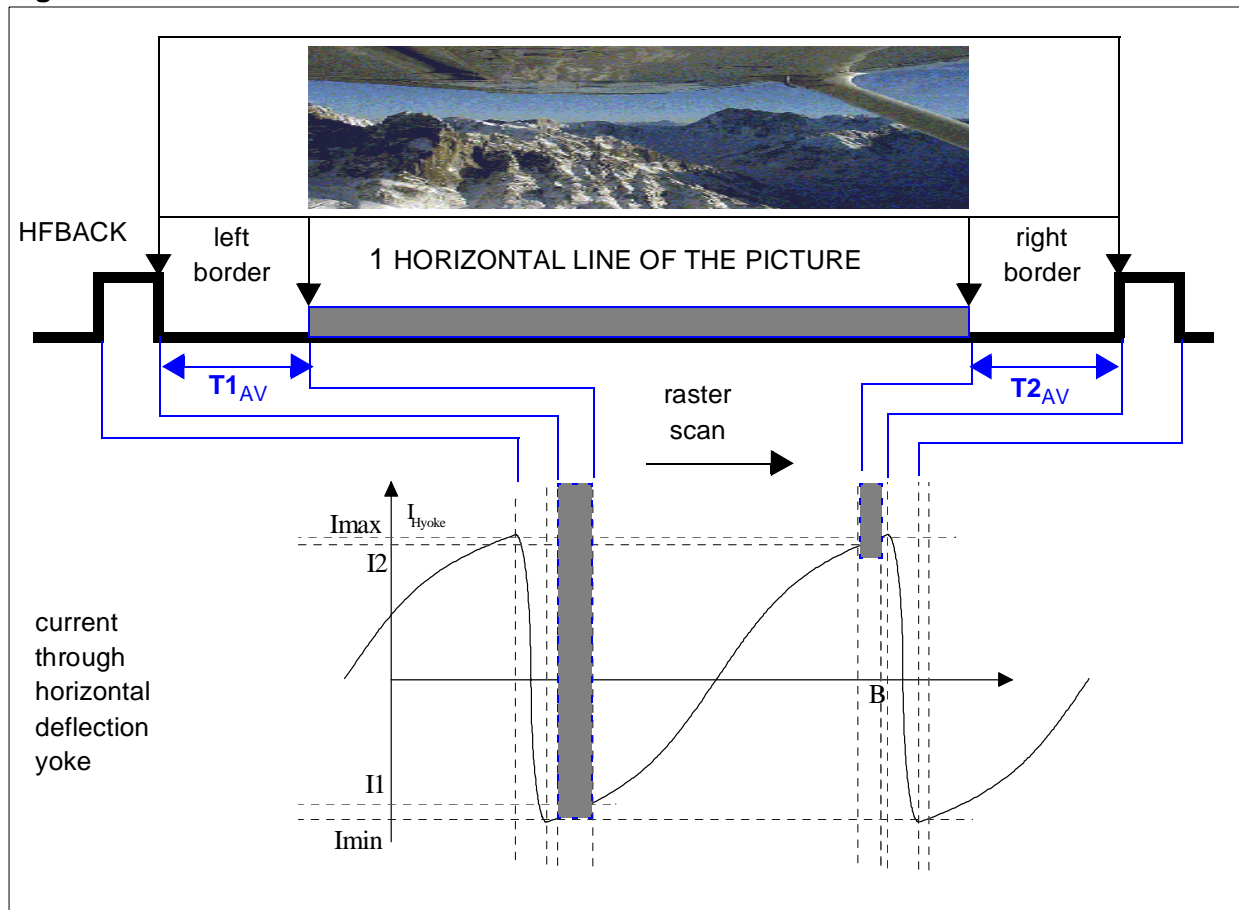
2.2 Horizontal sizing

In the case of horizontal sizing, the HFBACK pulse is useless because the time between 2 consecutive pulses remains the same, regardless of the width of the displayed picture: the size cannot be estimated that way.

A thorough study of the entire horizontal deflection stage allows the determination of the raster scanning curve. Usually, it is very difficult to determine a real mathematical equation for that curve. Actually, the horizontal deflection stage is not generated by a dedicated circuit but by several discrete components.

Furthermore, many coils (deflection, linearity..) and capacitors (S-correction..) are turned on and off depending on the horizontal frequency of the displayed picture, to compensate for the intrinsic limitations of the CRT. A curve model can be estimated only for a given frequency range.

Figure 5. Horizontal Raster Scan



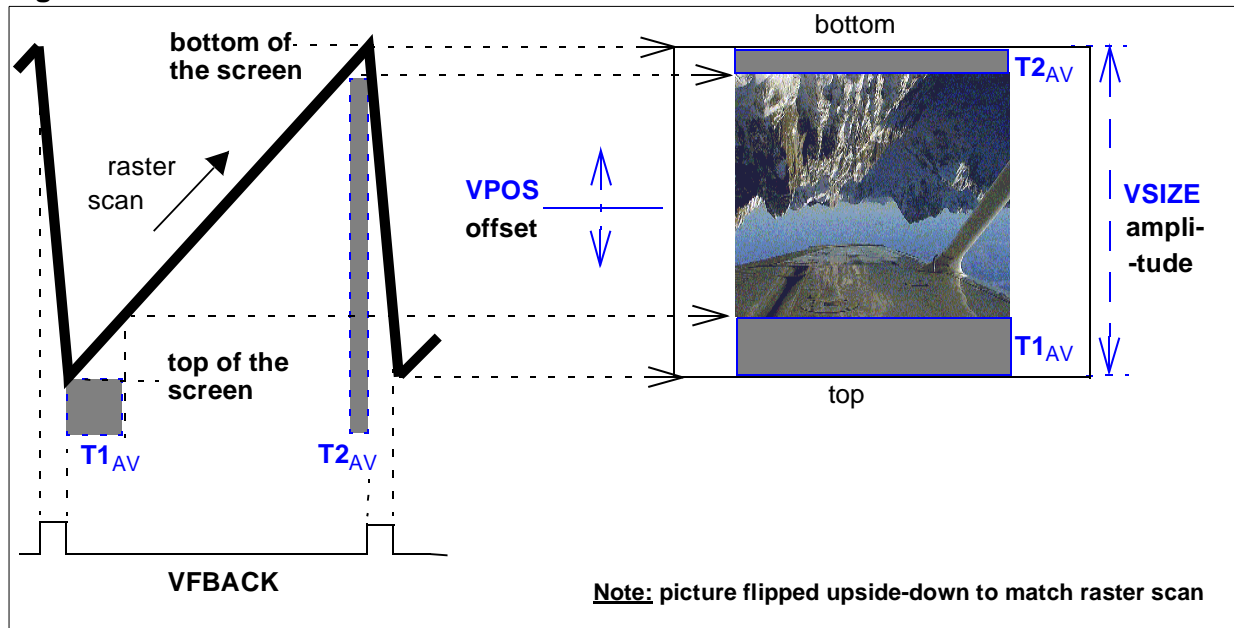
2.3 Vertical centering and sizing

Again, the horizontal centering principle does not apply to the Vertical Flyback pulse **VFBACK** because:

- The time between the top & bottom of the screen raster and the VFBACK pulses remains constant, regardless of the vertical position of the displayed picture : the centering cannot be estimated that way
- The time between 2 consecutive VFBACK pulses remains the same, regardless of the height of the displayed picture : the size cannot be estimated that way either

A thorough study of the entire vertical deflection stage is needed to determine a mathematical equation for the raster deflection curve. Thankfully, most datasheets for deflection processors (such as **TDA 9109** from STMicroelectronics) used in the monitor circuitry present the precise curve of the vertical deflection outputs (sawtooth) driving the vertical yoke.

Figure 6. Vertical Sawtooth



Once this curve is determined, the influence of the user settings **VSIZE** and **VPOS** over the curve needs to be computed. Those parameters are usually adjusted through I2C bus commands sent from the microcontroller to the deflection processor.

Finally, this leads to an equation defining the sawtooth voltage output as a function of :

- time (and vertical frequency)
- **VSIZE** and **VPOS** settings
- parameters $T1_{AV}$ and $T2_{AV}$ related to the displayed picture (as shown on [Figure 6](#))

3 THE NEW TMU CELL

3.1 Principle

The purpose of the Timing Measurement Unit is to measure the time values $T1_{AV}$ and $T2_{AV}$ allowing all the adjustments described above ; $T1_{AV}$ and $T2_{AV}$ depend on the displayed picture:

- $T1_{AV}$ is the time between the falling edge of the signal (synchro or flyback) and the very first non-black pixel of the picture : this measures the duration of the **left (H) or top (V) black border**
- $T2_{AV}$ is the time between the very last non-black pixel of the picture and the rising edge of the signal (synchro or flyback) : this measures the duration of the **right (H) or bottom (V) black border**

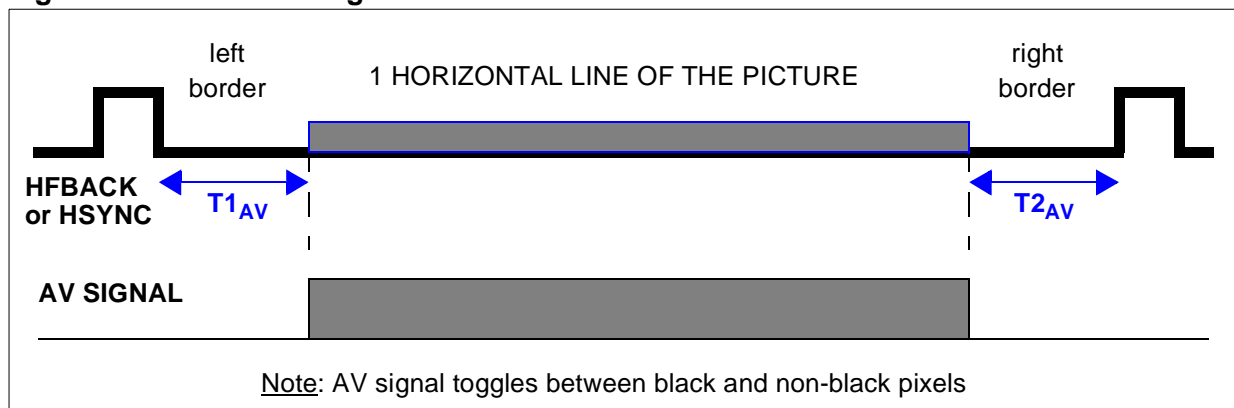
For a more detailed description of the cell, please refer to the corresponding TMU chapter in the ST72774 datasheet.

3.2 Active video signal

In order to distinguish between the displayed picture and the black borders surrounding the picture, an external signal called **ACTIVE VIDEO (AV)** is supplied to the TMU cell :

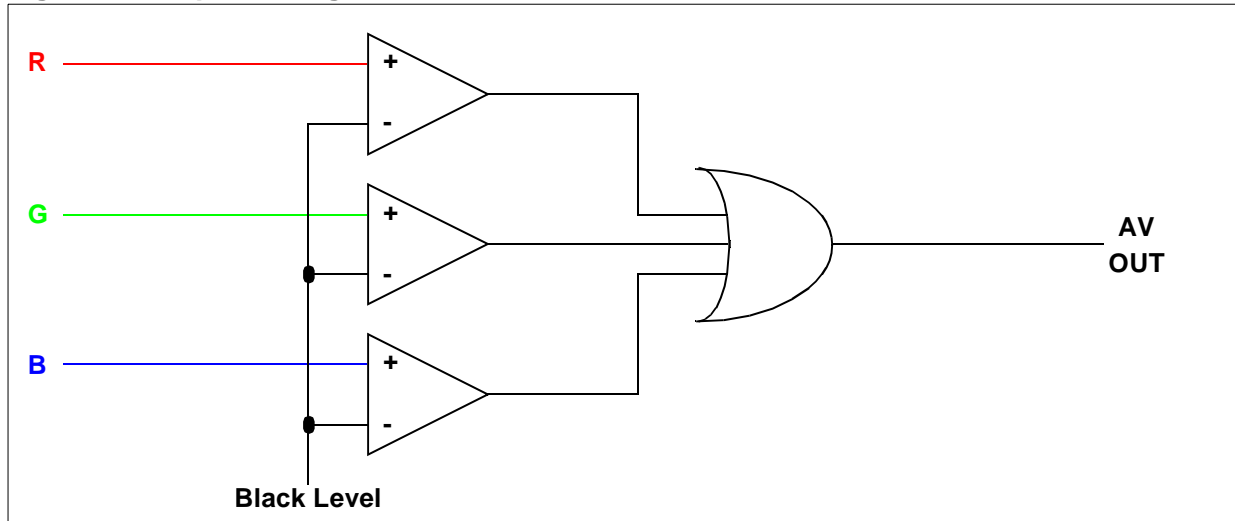
- AV is high (active) when any non-black pixel of the picture is displayed, that is when any of the three R G B colour signals is higher than the black level
- AV is low when no picture is displayed (nothing but black), that is when all three RGB colour signals are at black level

Figure 7. External AV Signal Generated over one Horizontal Line



The AV signal is therefore a simple OR a combination of three comparator outputs :

Figure 8. Simple AV Signal Generation



Another way to generate the AV signal is to use the video controller **TDA 9209** from STMicroelectronics. It has a built-in AV signal generator (up to 100MHz pixel clock) which is enabled/disabled by means of simple I²C commands.

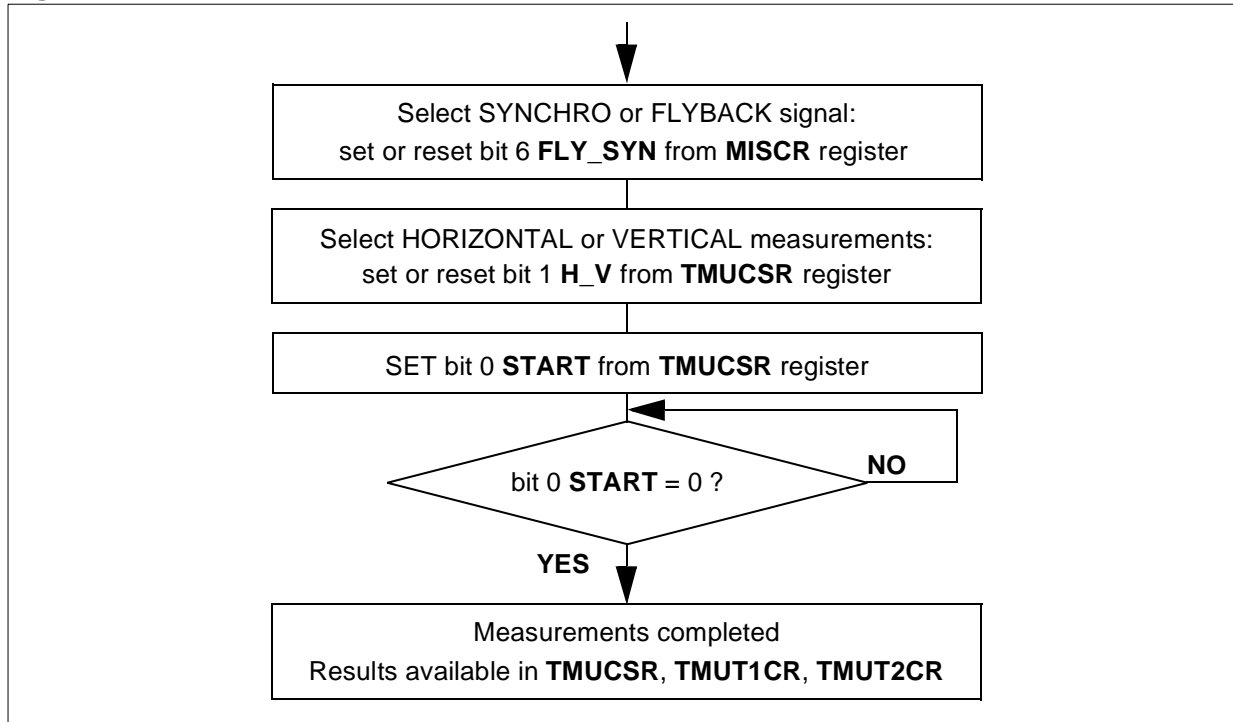
3.3 Start a measurement

A simple step-by-step procedure starts a given measurement:

- 1 Select reference signals: the measurements are either based on synchro signals HSYNC and VSYNC, or on flyback signals HFBACK and VFBACK. Selection is made with a bit **FLY_SYN** (#6) found in **MISCR** register
- 2 Select HORIZONTAL or VERTICAL signal : bit **H_V** (#1) of **TMUCSR** register
- 3 Start the measurements : this is done by setting bit **START** (#0) of **TMUCSR** register
- 4 Wait for measurements completion: wait until bit **START** (#0) is back to zero

The 11-bit values of **T1_{AV}** and **T2_{AV}** can then be read from registers **TMUCSR**, **TMUT1CR** and **TMUT2CR**.

Figure 9. Measurement Flow



3.4 Limitations and constraints

3.4.1 Overflow

If $T1_{AV}$ and $T2_{AV}$ values exceed the maximal value of $7FF_h$ ($2^{11} - 1$), an overflow occurs. In that case, the returned value will be $7FF_h$. This happens when the displayed picture is unusually small (very few pixels in size), like a DOS window with only “C:>” prompt.

3.4.2 The Active Video signal is too early

If for any reason, the first rising edge of the Active Video signal occurs **BEFORE** the falling edge of the reference sync signal, the returned value for $T1_{AV}$ is 1.

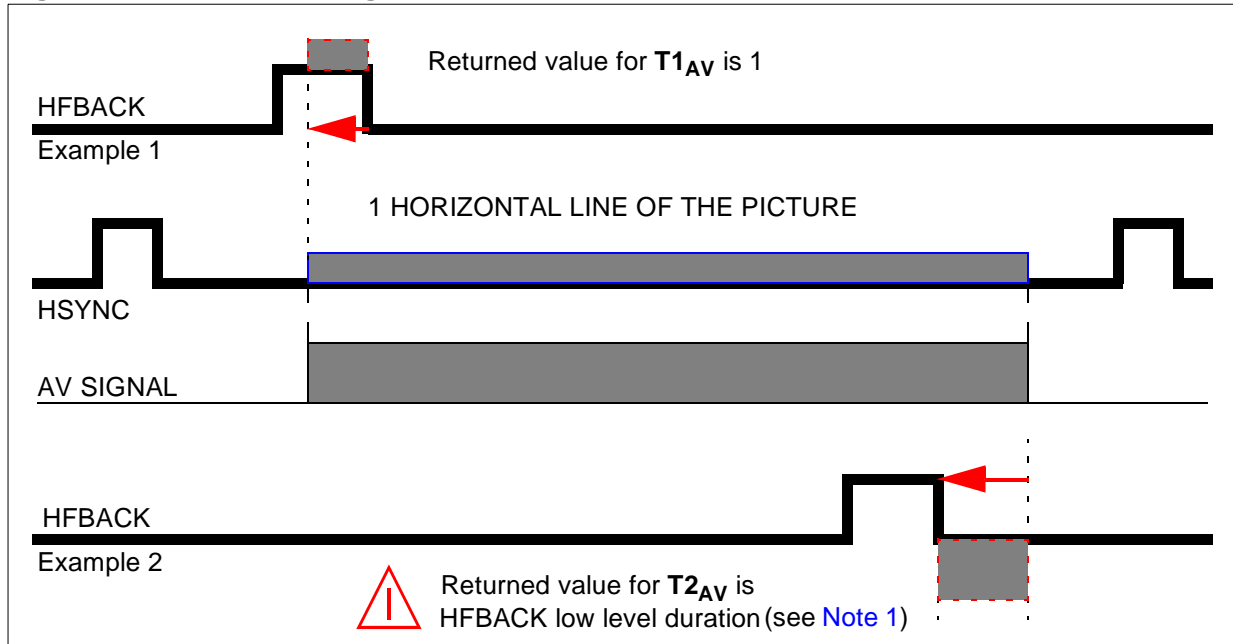
This happens while performing an horizontal measurement, when the reference signal is HFBACK and the picture is shifted too far to the left. In that case, it is advised to use HSYNC as reference instead of HFBACK.

3.4.3 The Active Video signal is too late

If for any reason, the last falling edge of Active Video signal occurs **AFTER** the rising edge of the reference sync signal, the returned value for $T2_{AV}$ is the low level duration of the signal pulse (time between the falling and rising edges) : see the note below.

This happens while performing an horizontal measurement, when the reference signal is HFBACK and the picture is shifted too far to the right. In that case, it is advised to use HSYNC as reference instead of HFBACK.

Figure 10. Unusual AV Signals



Note 1: The returned value for T2_{AV} is fixed by design and may change to 1 in the future.

3.4.4 Unstable picture

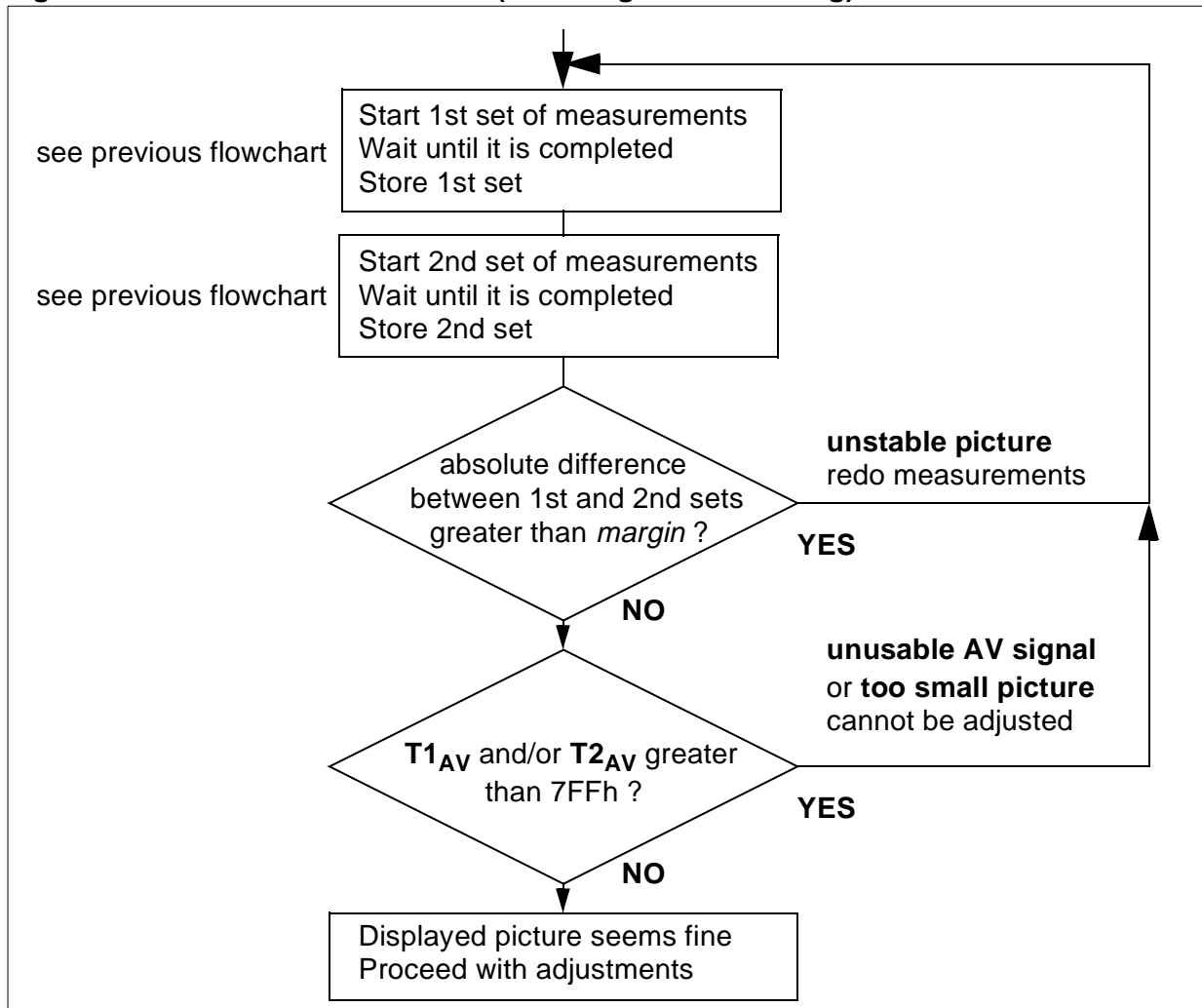
In case of a notably unstable picture, for example due to noise or when displaying an object in motion on a black background, the returned values for T1_{AV} and T2_{AV} do not remain constant across several consecutive frames.

For obvious reasons, autocentering and autosizing cannot be performed on pictures which are always moving on screen.

It is therefore strongly advised to perform at least **TWO** consecutive measurements based on the same reference sync signal, then to decide not to do any further adjustment if the measurements differ by more than a given margin.

Experiments showed that a margin of 5 clock pulses seemed safe enough.

Figure 11. TMU Measurement Flow (including error checking)



3.4.5 Measurement-resolution limitation

The values of $T1_{AV}$ and $T2_{AV}$ are 11-bit multiple of one internal clock period (either 41ns or 83ns, depending on the external XTAL following the oscillator clock f_{osc}). A simple example shows that, in very high resolution modes, it covers many pixels at a time:

- example video mode : 1280 x 1024, $fV = 74.112\text{Hz}$, $fH = 78.855\text{kHz}$
- horizontal scan line : total time = $1/fH = 12.681\mu\text{s}$, active video time = $9.481\mu\text{s}$
- 1280 horizontal pixels in $9.481\mu\text{s}$ corresponds to 1 pixel every 7.4ns
- TMU resolution at maximum speed ($f_{osc} = 24\text{MHz}$): 41ns == roughly **5 pixels step**

In that case, the best horizontal measurement precision of the TMU cell will be 5 pixels.

It is therefore strongly advised to use an external XTAL of the highest possible frequency, to obtain the finest resolution when doing measurements with the TMU.

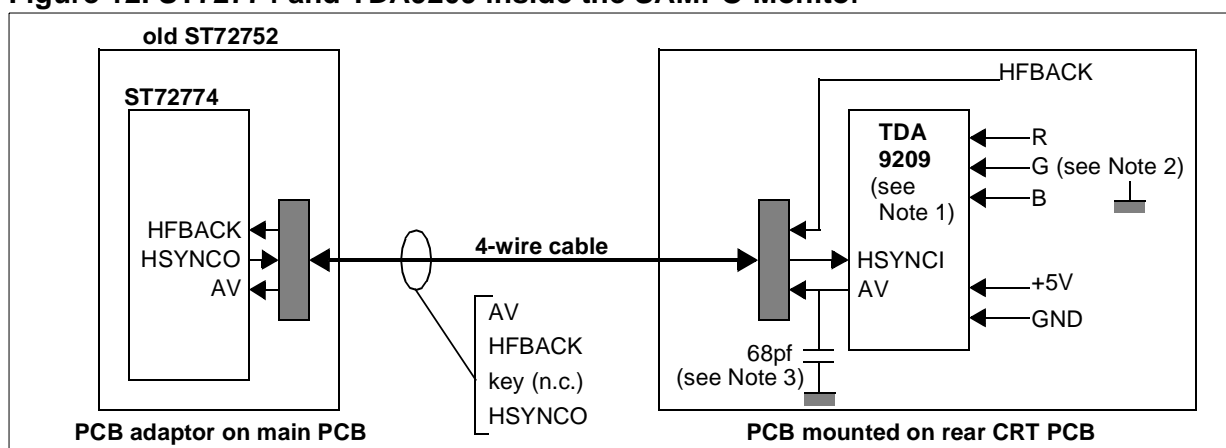
4 HARDWARE AND SOFTWARE IMPLEMENTATION IN A REAL MONITOR

4.1 H/W and S/W prerequisites

The monitor that we chose for our tests was the "KM751", a 17-inch CRT monitor display from Taiwanese manufacturer SAMPO. STMicroelectronics developed the whole software in partnership with SAMPO, in assembly language, for the ST72752 MCU.

The first step was to rewrite this software (and slightly work on the hardware around) to fit with the new ST72774 MCU. In a second step, a small additional PCB was designed to generate the required Active Video signal : this extra piece of hardware featuring the **TDA 9209** is mounted on the rear PCB at the back of the CRT, where all necessary signals are found (R G B and HFBACK) then passed on to the remote MCU.

Figure 12. ST72774 and TDA9209 inside the SAMPO Monitor



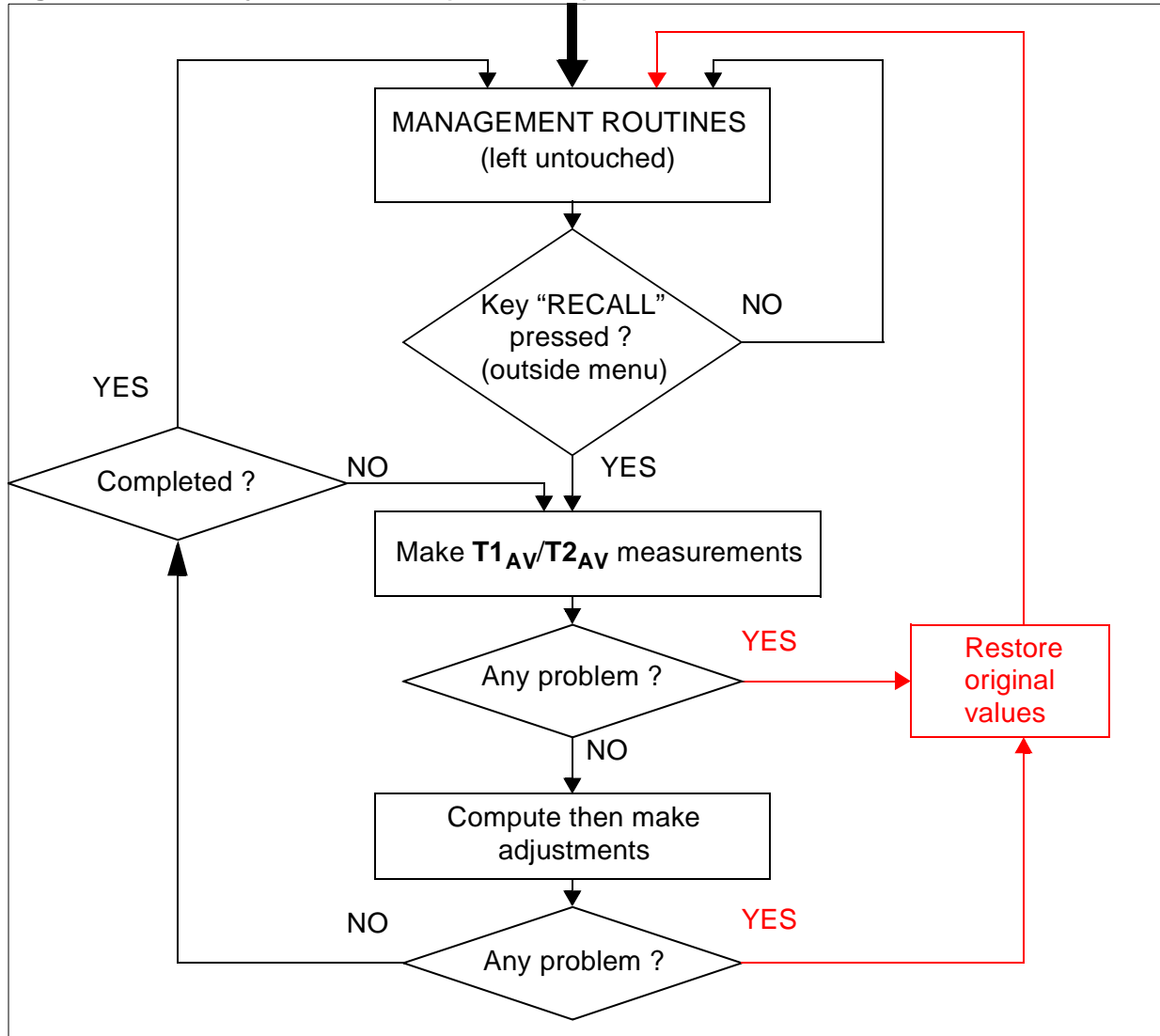
Note 1: The TDA9209 requires 3 **compulsory** I²C commands to configure its Active Video feature: [S DC_h 0D_h 8F_h P] to enable the AV output, then [S DC_h 08_h 90_h P] to configure the blanking signal, and finally [S DC_h 02_h 00_h P] to set the brightness register to zero.

Note 2: Used in this configuration (the chip not being implemented in the whole chassis but only as an add-on), the TDA9209 generates a bad AV signal when its Green input conveys a Sync-On-Green signal. This input should therefore be tied to ground for more reliable performance.

The original "RECALL" key has been slightly diverted : once pressed, when no menu is being displayed, it runs all the autosizing & autocentering routines. A message is then displayed while the adjustments take place. The displayed picture is also moved around during this time, this is a temporary but normal behaviour.

If the picture does not meet all the autoadjustment criterions (cf [Section 3.4 Limitations and constraints](#)), because it is unstable, in motion, too small to adjust, or it makes the adjustments fail for any possible reason, both former position & size settings will be restored.

Figure 13. Autoadjustment Flow (User Side)



4.2 Overall S/W behaviour

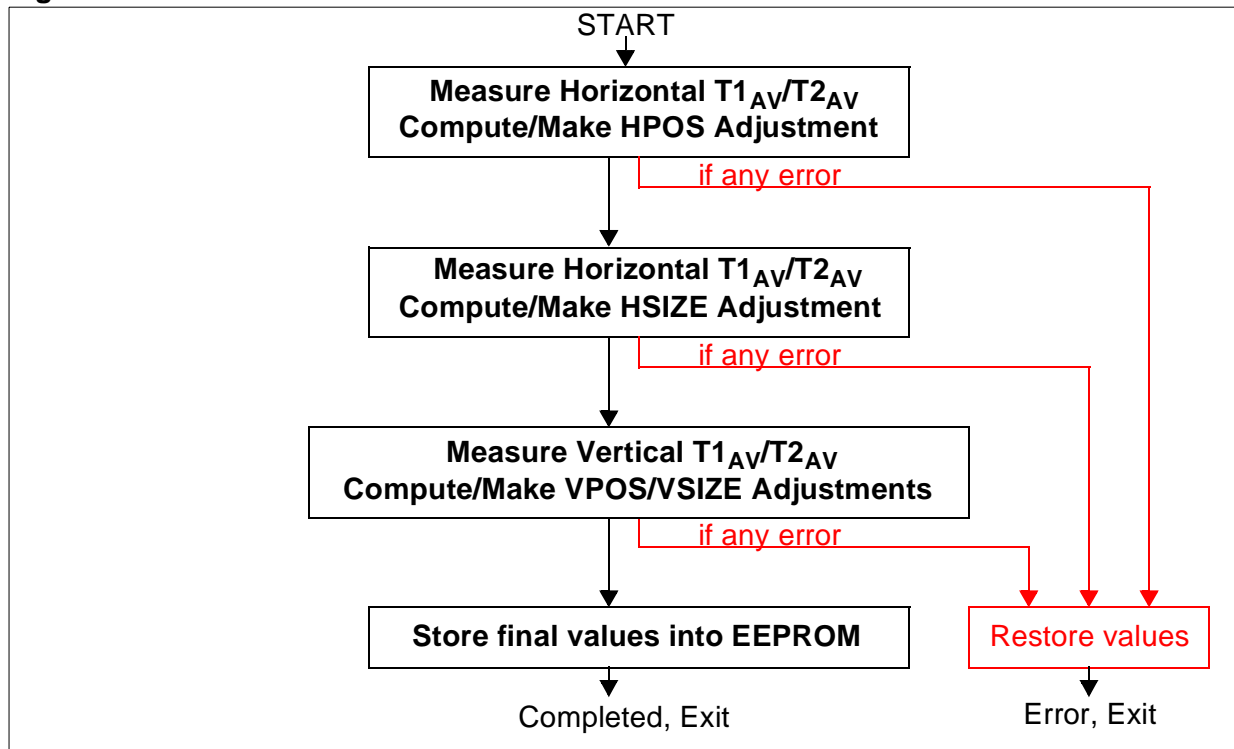
The S/W operates with the successive actions :

- 1 detect if the key "RECALL" has been pressed
- 2 start with HPOS (horizontal centering) adjustment
- 3 proceed with HSIZE adjustment (horizontal sizing)
- 4 proceed with both VPOS/VSIZE (vertical centering & sizing) adjustments
- 5 store the final adjusted parameters into monitor EEPROM, so that every time the same video mode is displayed, it comes up already auto-adjusted
- 6 in case of any error, initial values for HPOS/HSIZE and VPOS/VSIZE settings are restored, autoadjustment routines are aborted, and normal screen operation resumes

Theoretically, the 4 different adjustments are made in any order. However, it is advised to start with horizontal centering (HPOS), because a well-centered picture gives better results for horizontal sizing (HSIZE).

The other reason for our choice was a *logical* order, from the easiest to implement (HPOS) to the most complicated one (HSIZE and VPOS/VSIZE).

Figure 14. Overall S/W Flow



4.3 Horizontal auto-centering (HPOS)

Horizontal auto-centering is the simplest adjustment to implement because the position of consecutive HFBACK pulses against the AV signal changes with HPOS setting.

There are actually two ways to perform horizontal centering :

- One that runs a recursive algorithm which achieves the centering by successive approximations. It works in all cases but is slow by nature.
- One that performs 3 measurements of T1 and T2 then computes the final centering setting. It is extremely fast but may not work in all cases.

Those two algorithms, with their pros and cons, will be discussed respectively in [Section 4.3.1](#) and [Section 4.3.2](#).

4.3.1 Successive Approximations Algorithm

The screen is first shifted to a medium horizontal position, then shifted to the left or to the right step by step, by incrementing or decrementing the horizontal position setting, according to the difference between T1 and T2.

4.3.1.1 Computation time : How to keep it minimum

Before making the proper adjustment, the time required to adjust the picture must be taken into consideration. The following example is studied in details : VGA mode 640x480 pixels, fV=60Hz, fH=31.5kHz.

Each measurement pass takes 2 vertical synchro pulses (as stated in the TMU cell datasheet) to complete. Considering fV=60Hz, this time is about 17ms. **T1_{AV}** then **T2_{AV}** are measured. Another measurement pass is then performed (cf [Section 3.4](#)). This means that for 2 consecutive HPOS adjustments, it takes about **34ms** at fV=60Hz.

On the SAMPO screen, the HPOS setting varies between 0 (leftmost) and 127 (rightmost), which makes 128 steps in total. The average setting to get an approximately centered picture in all video modes is half of the maximum value, 64.

If the picture is completely shifted outside the screen to the left or to the right, an average of 64 HPOS setting steps (half of the total HPOS range) is needed to shift the picture towards the appropriate direction. Each step lasting 34ms, the total adjustment process may take about $64 \times 34 = \mathbf{2.2 \text{ seconds}}$.

To decrease this duration, the HPOS setting is advised to be preset to its half-range value (64 in this case) before doing any adjustment. That way, the picture is already roughly centered, or at least not shifted too far from the center of the screen. Therefore, only a few further steps are needed to center the picture perfectly.

4.3.1.2 Measurements of T1_{AV}/T2_{AV} and preliminary tests

A 1st set of **T1_{AV}/T2_{AV}** measurements is made, followed by a 2nd set. Both sets are then compared to each other: if their absolute difference is greater than a fixed value (TMUDIFF), the picture is considered to be unstable or in motion, therefore the routine exits and restores the original HPOS value. The routine also aborts if **T1_{AV}** and/or **T2_{AV}** value equals the maximum value (7FF_h) meaning the picture is too small to be adjusted, or the AV signal is wrong.

For the complete algorithm, refer to [Figure 11](#)

Note : The above preliminary tests are common to all adjustments.

4.3.1.3 UPDOWN flag setting

For the very first set of measurements, **T1_{AV}** is compared to **T2_{AV}** to determine the adjustment of the picture position, as described in [Section 2.1](#):

AUTO-SIZING & AUTO-CENTERING

- if $T1_{AV} < T2_{AV}$, the picture is already shifted to the left and needs to be moved rightwards; this means increasing the actual HPOS setting (up to 127 max); UPDOWN flag =0
- if $T1_{AV} > T2_{AV}$, the picture is already shifted to the right and needs to be moved leftwards; this means decreasing the actual HPOS setting (down to 0 min); UPDOWN flag =1

This flag is kept in memory: it indicates which way the HPOS setting must be modified (incremented or decremented). It is not modified anymore.

4.3.1.4 HPOS setting update

From now on and until the picture is considered adjusted, the S/W performs different tests and actions depending on the value of the UPDOWN flag described previously:

- UPDOWN = 0: check if $T1_{AV}$ is still $< T2_{AV}$
 - **yes**: increment HPOS and update setting
 - **no**: the picture is now centered because last HPOS step made $T1_{AV} > T2_{AV}$ (opposite from initial condition)
- UPDOWN = 1: check if $T1_{AV}$ is still $> T2_{AV}$
 - **yes**: decrement HPOS and update setting
 - **no**: the picture is now centered because last HPOS step made $T1_{AV} < T2_{AV}$ (opposite from initial condition)

If either answer was “**yes**”, the routine loops back to the 1st and 2nd sets of $T1_{AV}/T2_{AV}$ measurements at the very beginning. Otherwise, the picture is now assumed to be centered, the routine is completed.

Note : A short delay is necessary for the screen to take the setting update into account. This is due to the normal latency of the internal monitor circuitry when there is a change in any setting. A delay of 10ms was tested to give good results. This also applies to preliminary HPOS preset (refer to [Section 4.3.1.1](#)).

4.3.1.5 Avoid “never-ending” loops

Every time HPOS setting is modified, a counter is incremented then compared to a maximum number of iterations (*TMURETRY*). If the counter exceeds this maximum number, the routine, which is then unable to adjust HPOS, restores the original setting and aborts.

This is an additional safety measure against possible infinite loops within the routine.

4.3.1.6 Complete HPOS algorithm

Called from the main loop whenever the “RECALL” key is pressed, the HPOS adjustment loop shows two normal exit conditions (when the picture is found centered on screen) and three abnormal exit conditions (when the picture is not stable, unusable, or cannot be adjusted within a given number of trials).

Figure 15. Complete HPOS algorithm

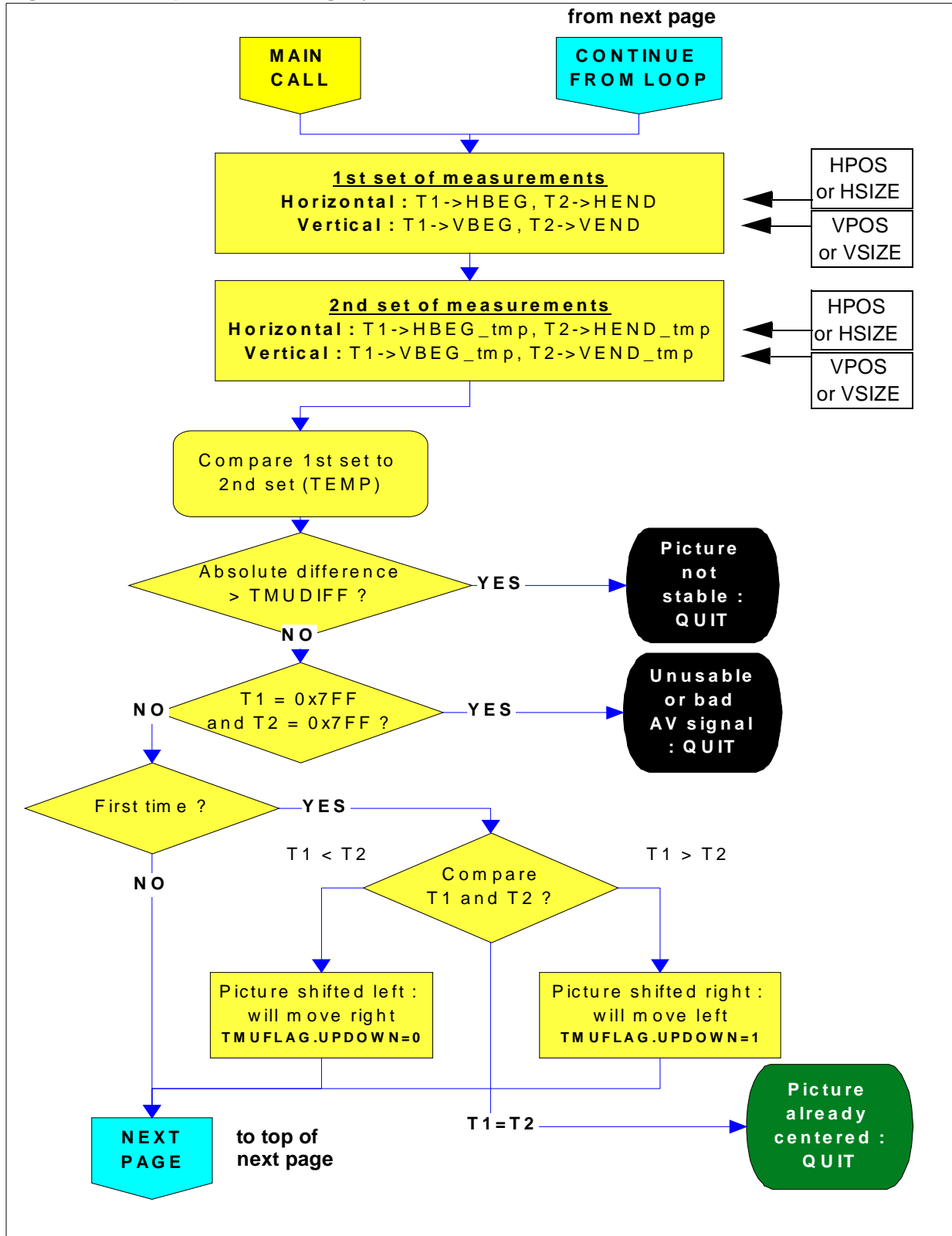
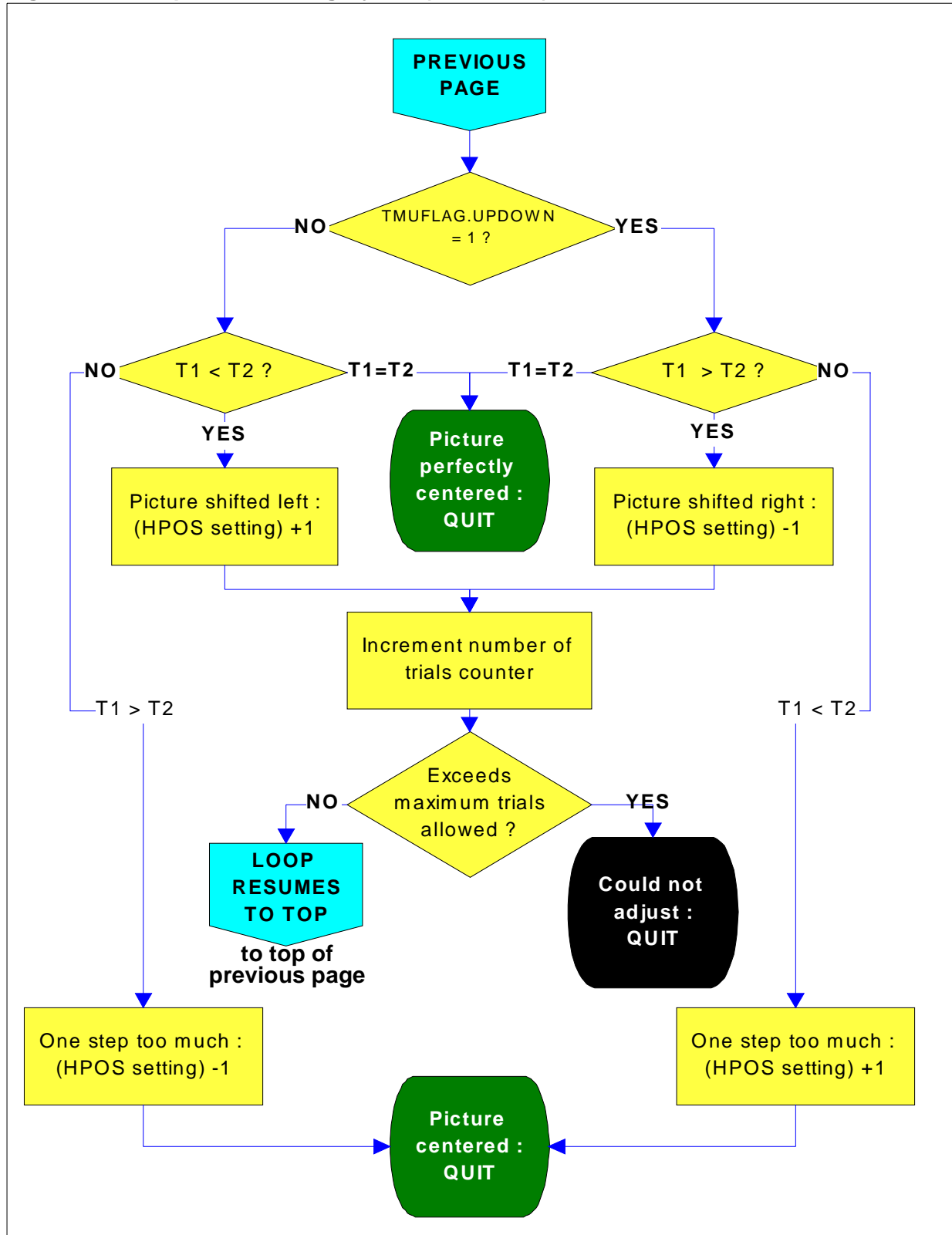


Figure 16. Complete HPOS algorithm (continued)



4.3.2 Measurements & Computation Algorithm

Three measurements are performed to estimate the optimum value and the max/min for $T1=T2$, then the final horizontal position setting is computed by extrapolation.

4.3.2.1 First measurement : rough centering

As in [Section 4.3.1.1](#), the HPOS setting is first preset to its half-range value which should roughly center the displayed picture. This is enough to measure $T1$ and $T2$ then compute the average of the two :

$$T2_{opti} = \frac{(T1 + T2)}{2}$$

This average value $T2_{opti}$ is the optimum $T1=T2$ to achieve for optimum horizontal centering, since the sum of back and front porches ($T1+T2$) remains constant across one line, no matter how shifted the line is.

Important Note : if the first measurement returns a bad (out of range) value for $T1$ and/or $T2$, due to the fact that a medium value for HPOS setting may not always put the displayed picture right between 2 consecutive HFLYBACK pulses, the final computed HPOS value will be completely wrong. For that matter, it is safer to use HSYNC as reference signal instead, and the displayed picture is then guaranteed to always fit between 2 consecutive HSYNC pulses.

4.3.2.2 Second measurement : left shift

Next, the HPOS setting is set to its minimum value (usually 0) so that the displayed picture is completely shifted to the left. As such, the newly measured $T2_{max}$ is the maximum achievable value for $T2$ (front porch maximum).

4.3.2.3 Third measurement : right shift

Next, the HPOS setting is set to its maximum value (usually 100 or 255) so that the displayed picture is completely shifted to the right. As such, the newly measured $T1_{max}$ is the maximum achievable value for $T1$ (back porch maximum).

4.3.2.4 Optimum centering computation

Next, the optimum horizontal centering setting HPOS is computed by linear extrapolation from $T2_{opti}$, $T2_{max}$ and $T1_{max}$ which were obtained at previous steps.

Considering that $(T1+T2)$ remains constant, from $T1_{max}$ we can compute ::

$$T2_{min} = T1_{max} - (T1 + T2)$$

AUTO-SIZING & AUTO-CENTERING

Then considering that $T2$ is maximum ($=T2_{max}$) for $HPOS=0$ and minimum ($=T2_{min}$) for $HPOS_{MAX}$, we can do a linear extrapolation to compute the optimum horizontal centering value $HPOS$ for $T2=T2_{opti}$ as computed in [Section 4.3.2.1](#) above :

$$T2_{opti} = T2_{max} - (T2_{max} - T2_{min}) \times \frac{HPOS}{HPOS_{max}}$$

Reversing the formula gives $HPOS$ from $T2_{opti}$:

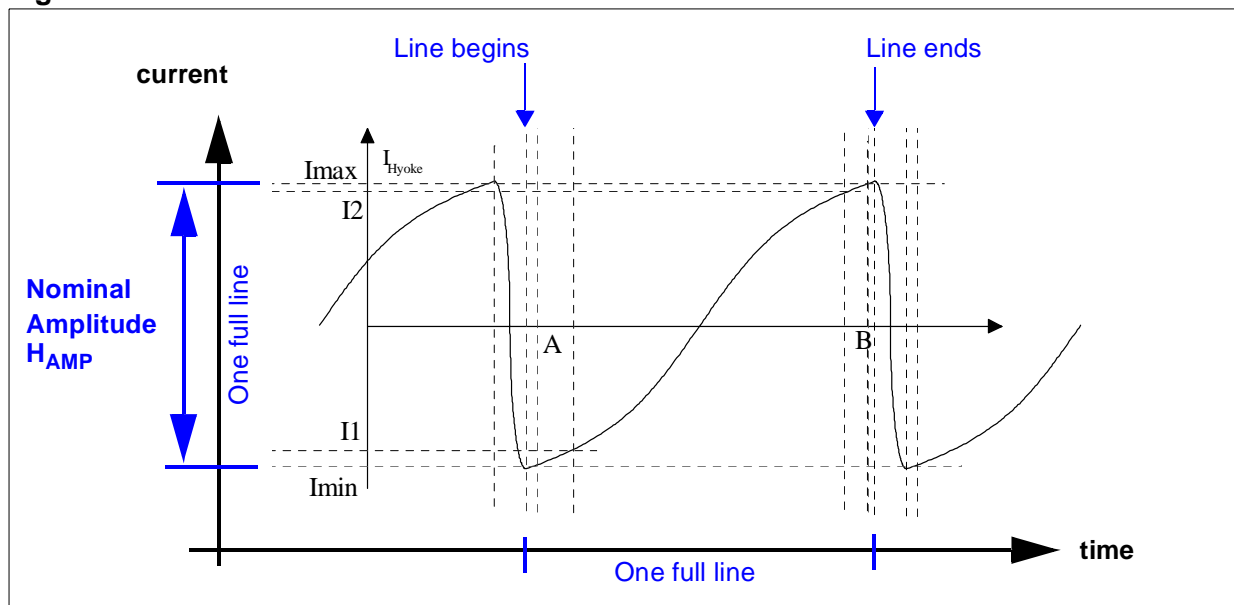
$$HPOS = \left(\frac{T2_{max} - T2_{opti}}{T2_{max} - T2_{min}} \right) \times HPOS_{max}$$

4.4 Horizontal autosizing (HSIZE)

The horizontal spot deflection is based on the principle of a variable current flooding through the horizontal deflection coil (since a magnetic field inside a coil is proportional to the current running through it). But this is also the most intricate parameter to compute as:

- there is no on-chip fixed waveform generated by any deflection processor circuit
- the horizontal deflection current waveform is very complex (by nature)
- the shape of the sawtooth current waveform changes as the horizontal frequency varies with the addition/removal of various coils and capacitors in serial and/or parallel with the horizontal deflection coil (in order to make some corrections)

Figure 17. Horizontal Deflection Curve



4.4.1 Horizontal deflection formulas

According to preliminary measurements, the overall amplitude of a line, H_{AMP} , varies with the horizontal frequency f_H of the video mode being displayed, but also with **HSIZE** setting.

Moreover, certain adjustments are made by S/W by **range of horizontal frequencies**. This is needed for linearity and S-correction reasons, to name a few.

Therefore, a first step is to find a relationship between f_H and the measured amplitude of the current H_{AMP} for each range of horizontal frequencies. This leads to several different formulas:

$$H_{AMP} = f(F_H)$$

This relationship is first assumed to be **linear**, in the form of (ax+b). This makes later computations much easier to implement, but it requires some corrections that are discussed later on.

The next problem is that the above formula does not remain constant as HSIZE setting itself varies between its minimum **HSIZE=H_{MIN}** (shrunk line) and maximum **HSIZE=H_{MAX}** (enlarged line) values : linear factors “a” and “b” appear to vary linearly with HSIZE setting.

In a 2nd step, a separate formula should then be established for each of the 2 extremum HSIZE values above:

$$H_{AMP_{MIN}} = f(F_H)_{HSIZE=H_{MIN}}$$

..and :

$$H_{AMP_{MAX}} = f(F_H)_{HSIZE=H_{MAX}}$$

In the case of SAMPO monitor, HSIZE setting adjusts PWM output #2 (signal DA2 pin 2 of ST72752), therefore $H_{MIN}=0$ and $H_{MAX}=255$.

Assuming that H_{AMP} varies **linearly** along with HSIZE setting, the 2 above formulas can be interpolated to give one single linear formula :

$$H_{AMP(HSIZE)} = f(F_H, HSIZE)$$

.. and finally:

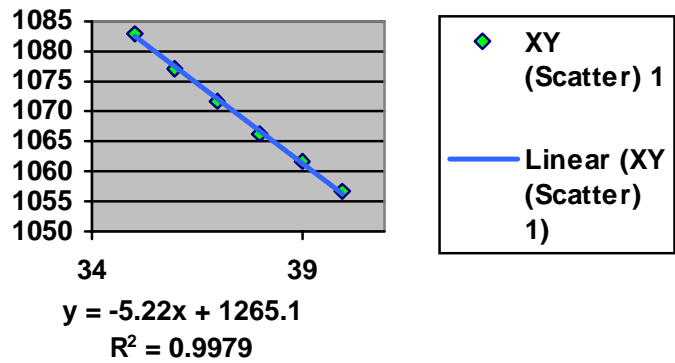
Figure 18. Horizontal Amplitude, function of HSIZE setting

$$H_{AMP(HSIZE)} = H_{AMPMIN} + (H_{AMPMAX} - H_{AMPMIN}) \times \frac{HSIZE}{HSIZE_{MAX}}$$

Note : There is one $H_{AMP}(HSIZE) = f(f_H)$ formula for each range of horizontal frequencies.

This curve is an **example** of $H_{AMPMIN}=f(f_H)$ on a SAMPO monitor, in $f_H = [34kHz...41kHz]$ range :

- **x** is f_H (in kHz)
- **y** is $H_{AMP} \times 10mA$ (measured with a 10mV/mA current clip probe)
- **R** is the linear regression factor, very close to 1 (= almost perfect linear interpolation)



Note: refer to [Section 7 ANNEX](#) for all SAMPO curves

4.4.2 Amplitude formulas

The above H_{AMP} formula computes the total (nominal) amplitude of one scan line.

However, this nominal amplitude must be “scaled down” to compute the line amplitude of the real picture (during active video) : A_V .

A_V can be expressed as a fraction of the total line amplitude H_{AMP} scaled by the ratio between the active video line time T_d (real picture only, without black borders) and the total line duration T :

$$A_V = H_{AMP} \times \left(\frac{T_d}{T}\right)$$

H_{AMP} being the total picture line size, the “active” line amplitude decreases as the black borders of the picture increase ($T_d \ll T$). Conversely, as the “active” line size gets larger and closer to the nominal line amplitude, its amplitude increases ($T_d \approx T$).

4.4.3 Operating mode

A given video mode is displayed. This reference picture is adjusted to best fit the screen horizontally (smallest possible left/right black borders), and its optimum current amplitude A_{Vopti} corresponding to the scan line is measured with a scope.

This A_{Vopti} becomes the reference-optimum-amplitude to reach in all video modes. It is expected to remain constant over all f_H ranges : a certain constant amount of current applied to the deviation coil sweeps a constant scan line area on screen.

• In case of SAMPO monitor : $A_{Vopti \text{ measured}} = 1130$ (x10mA, see example in [Section 4.4.1](#)).

When a new video mode is displayed, its new T_d and T values must be measured.

Then its overall amplitude H_{AMP}' needs to be adjusted so that its AV amplitude A_V' coincides with the reference-optimum amplitude A_{Vopti} measured previously:

$$A_V' = A_{Vopti} = H_{AMP}' \times \left(\frac{T_d'}{T'} \right)$$

finally:

Figure 19. Horizontal Amplitude, function of T' and Td'

$$H_{AMP}' = A_{Vopti} \times \left(\frac{T'}{T_d'} \right)$$

H_{AMP}' depending on f_H and HSIZE setting, it is now easy to reverse the formula in [Section 4.4.1](#) and get the new $HSIZE'$ setting to obtain the best fitted picture.

4.4.4 Optimum HSIZE setting formula

Equalizing the two H_{AMP} formulae established in [Figure 18](#) and [Figure 19](#) leads to the following formula expressing HSIZE setting value as a function of all the other relevant parameters :

Figure 20. Optimum HSIZE Setting Formula

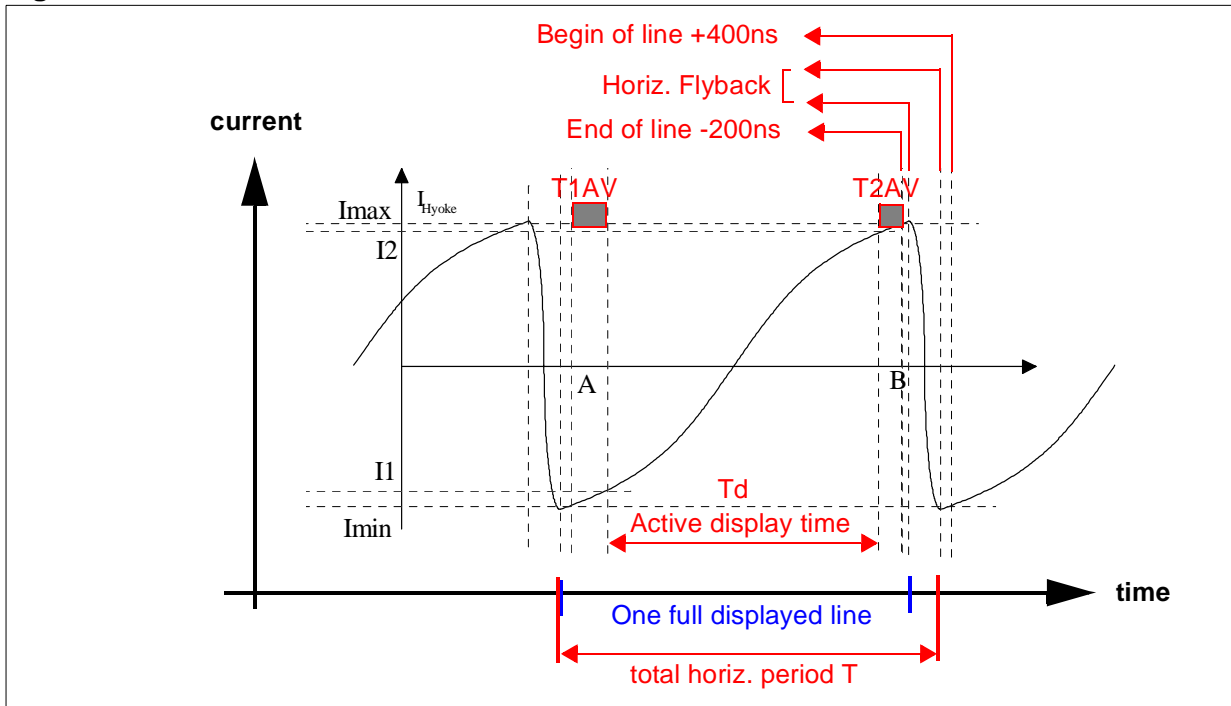
$$HSIZE = \frac{\left(A_{Vopti} \times \frac{T'}{T_d'} \right) - H_{AMP_{MIN}}}{H_{AMP_{MAX}} - H_{AMP_{MIN}}} \times HSIZE_{MAX}$$

Given $A_{Vopti \text{ measured}} = 1130$.

4.4.5 Measurement of T_d' and T'

Those 2 values are measured with the TMU cell.

Figure 21. All Horizontal Parameters of a Scan Line



On the sawtooth curve :

- n T_{1AV} is the leftmost black border (before the displayed picture)
- n T_{2AV} is the rightmost black border (after the displayed picture)
- n T_d is the active display time (during which a picture is really displayed)
- n $T_{flyback}$ is the horizontal flyback time
- n T is the total period of one horizontal line

In order to report them into the formula from previous [Section 4.4.3](#) , T_d' and T' are calculated as follows:

$$T_d' = T - (T_{1AV} + T_{2AV} + T_{FLYBACK})$$

and :

$$T' = T - \underbrace{(T_{FLYBACK} + 200ns + 400ns)}_{3.6\mu s \text{ in total}}$$

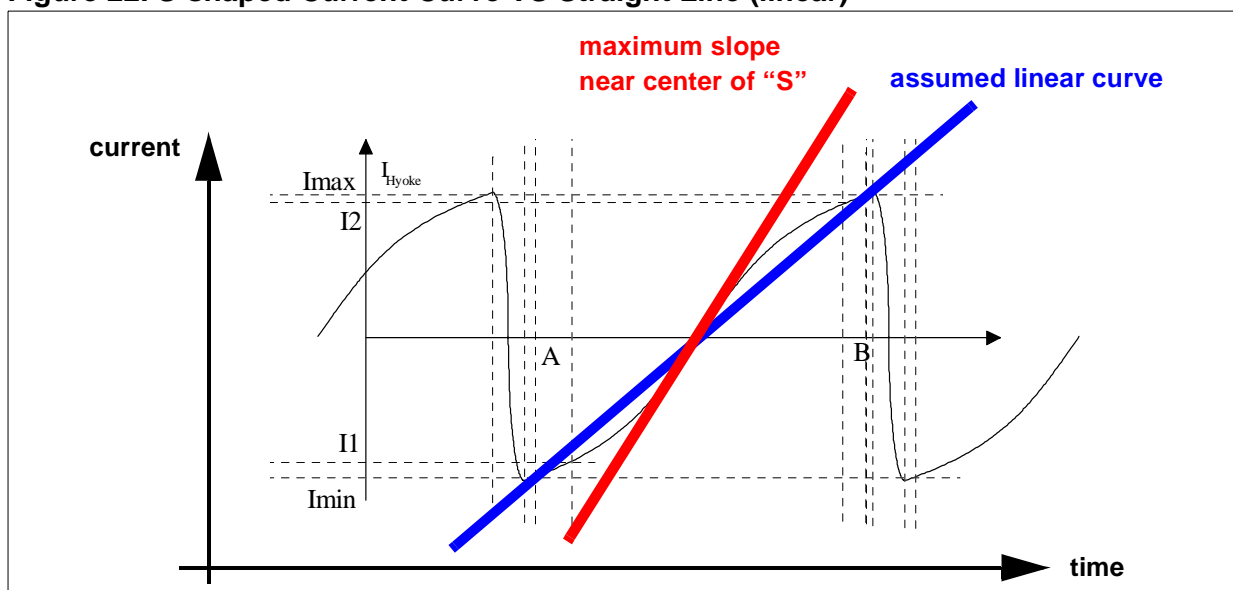
$T_{flyback}$ is constant (except at $f_H > 90kHz$) and equal to $3\mu s$.

As shown in Figure 21, “safety margins” of 200ns and 400ns are set *before* and *after* the flyback pulse respectively. This is due to the slew rate of the coil’s current curve $I=f(t)$ which is not sharp around the flyback rising and falling edges, in addition to some ringing. Consequently, the measurement of the most precise flyback duration is very inaccurate, therefore this “fuzzy” area is excluded from the measurement in all video modes.

4.4.6 Additional compensation parameter

In Section 4.4.1, the H_{AMP} current curve was assumed **linear** to make all the computations easier. In practice, with the formula given in Figure 20 the calculated HSIZE values are too high and tend to enlarge the picture beyond the horizontal CRT size.

Figure 22. S-shaped Current Curve VS Straight Line (linear)



The current deflection curve is not linear but S-shaped. Therefore, a relatively small picture, whose initial amplitude slope across a line is closer to the “S” curve than to a straight line, is enlarged more than needed.

The formula shown in Figure 18 must then be multiplied by a factor proportional to the T_d/T ratio. Experiments on several SAMPO monitors showed that the most fitted factor was :

$$\underbrace{\left(A_{Vopti} \times \frac{T'}{T_d'} \right)}_{\text{original formula}} \cdot \underbrace{\left(1 - \frac{1 - \frac{T_d'}{T'}}{2.8} \right)}_{\text{correction factor}}$$

AUTO-SIZING & AUTO-CENTERING

- If T_d is close to T , T_d/T is close to 1 : the picture is nearly optimum (near the actual size of the CRT), so there is less need of a correction, the above factor is close to 1 (no effect)
- If $T_d \ll T$, T_d/T is close to 0: the current amplitude is close to the center of the S-shaped curve which slope is at its maximum, the above factor is minimum $(1 - 1/2.8) = 0.64$

Note : This correction factor is written in a strictly identical manner:

$$\dots \cdot \left(\frac{1.8T' + Td'}{2.8T'} \right)$$

4.4.7 Final HSIZE formula

Once the correction factor is fed into the formula from [Figure 20](#), we have:

Figure 23. Final HSIZE Setting Formula

$$HSIZE = \frac{A_{Vopti} \cdot \left(\frac{1.8T' + Td'}{2.8Td'} \right) - H_{AMPMIN}}{H_{AMPMAX} - H_{AMPMIN}} \times HSIZE_{MAX}$$

Given $A_{Vopti \text{ measured}} = 1130$.

4.4.8 Software implementation

The result of all computations is restricted to unsigned 32-bit wide integer variables (*long int* type in C language). This is done by using 16-bit unsigned integer parameters. Care must be taken to avoid overflows and loss of precision.

For example, any division with a result < 1 must be avoided. Instead, the numerator should be the highest possible number, and the denominator the smallest, to minimize the rest (hence the *yyyy* part of the *xxxx.yyyy* result which, being integer, is ignored).

For this reason, several parameters from the numerator of a division are multiplied by a factor of 256. In the HSIZE formula, the final result of the division in [Figure 23](#) is a [0..1] factor of $HSIZE_{MAX} = 255$. Therefore, all the parameters from the numerator in [Figure 23](#) are multiplied by 256, then added or subtracted as needed.

Note : Values taken from the reference picture are underlined.

The current implementation for HSIZE is then:

- 1 Compute $T_{x256} - (T_{\text{flyback}} + 200\text{ns} + 400\text{ns}) \times 256$
- 2 Compute $T' \times 256 = [1] \times 9 / 5 = [1] \times 1.8$ and store it
- 3 Compute $(T_{1AV} + T_{2AV}) \times 256$, then /24 and store it
The result is divided by 24 because T_{1AV} and T_{2AV} are expressed in f_{CPU} pulses (1/24MHz) so they must be converted to micro-seconds ($\times 10^6 / 24 \times 10^6 == /24$)
- 4 Compute $T_{x256} - T_{\text{flyback}} \times 256$
- 5 Compute $T_d' \times 256 = [4] - [3]$ then store it
- 6 Compute $(1.8T' + T_d') \times 256 = [2] + [5]$
- 7 Compute $[6] \times (A_{\text{Vopti}} / 2.8)$
- 8 Compute $[7] / [5]$ then store it
- 9 Compute $f_H \times 256 = 1000 \times 256 \times 256 / (T_{x256})$, the result is in kHz $\times 256$
- 10 Load "**Ax100**" multiplier from COEFF1A Rom Table, compute $[9] \times (A \times 100)$
- 11 Compute $[10] / 100$ and store it
- 12 Load "**Bx256**" factor from COEFF1B Rom Table
A and **B** pertain to $H_{\text{AMPMIN}} = f_H \times A + B$
- 13 Compute $H_{\text{AMPMIN}} \times 256 = [12] - [11]$
a subtraction is performed because "A" factor is always negative in practice
- 14 Compute $[8] - [13]$ and store it : this is the numerator $\times 256$ of the formula in [Figure 23](#)
- 15 Compute $f_H \times 256 = 1000 \times 256 \times 256 / (T_{x256})$, the result is in kHz $\times 256$
- 16 Load "**Ax100**" multiplier from COEFF2A Rom Table, compute $[15] \times (A \times 100)$
- 17 Compute $[16] / (100 \times 256)$ and store it
- 18 Load "**Bx256**" factor from COEFF2B Rom Table
A and **B** pertain to $H_{\text{AMPMAX}} = f_H \times A + B$
- 19 Compute $H_{\text{AMPMAX}} = [18] - [17]$
- 20 Compute $H_{\text{AMPMAX}} - H_{\text{AMPMIN}} = [19] - ([13] / 256)$: this is the denominator of the formula in [Figure 23](#)
- 21 Compute $\text{HSIZE} = [14] / [20]$
There is no need to multiply HSIZE by $\text{HSIZE}_{\text{MAX}}$ (=255) because **[14]** is already $\times 256$
- 22 Compare **HSIZE** against maximum setting value (255) : if greater, use 255 instead
- 23 Send **HSIZE** setting to the deflection circuitry through PWM output DA2

4.5 Vertical autocentering/autosizing (VPOS, VSIZE)

4.5.1 Vertical sawtooth formula

The deflection processor used in the SAMPO monitor is the TDA 9109 in which datasheet the vertical ramp equation available on pin 23 is given:

Figure 24. TDA9109 VOUT formula (from datasheet)

$$V_{OUT} = V_{POS} + (V_{OSC} - V_{DCMID}) \times (1 + 0.25 \times V_{AMP})$$

Basic Equations

In first approximation, the amplitude of the ramp on Pin 23 (VOUT) is:

$$V_{OUT} - V_{POS} = (V_{OSC} - V_{DCMID}) \cdot (1 + 0.25 (V_{AMP}))$$

where:

$$V_{DCMID} = 7/16 V_{REF}$$

(middle value of the ramp on Pin 22, typically 3.5V)

$$V_{OSC} = V_{22} \text{ (ramp with fixed amplitude)}$$

$V_{AMP} = -1$ for minimum vertical amplitude register value and $+1$ for maximum

VPOS is calculated by:

$$V_{POS} = V_{DCMID} + 0.3 V_P$$

where $V_P = -1$ for minimum vertical position register value and $+1$ for maximum.

Note : All additional correction factors, like “Vertical S Linearity” and “Vertical C Linearity”, cannot be taken into account by such a simple formula. However, in practice, this gave pretty good results, because the correction factors represent less than 4% of the final V_{OUT} result.

VAMP and VPOS are controlled by “Vertical Size” (amplitude of the curve) and “Vertical Position DC Control” (DC offset of the curve) parameters respectively. The above formula is then simplified into a simple ramp-down slope defined by VAMP and VPOS:

$$V_{OUT} = V_{OUTDC} + V_{AMP} \times (0.5 - x)$$

This formula represents the simplest description of the sawtooth output curve. It is also simple to implement in assembly language in a microcontroller.

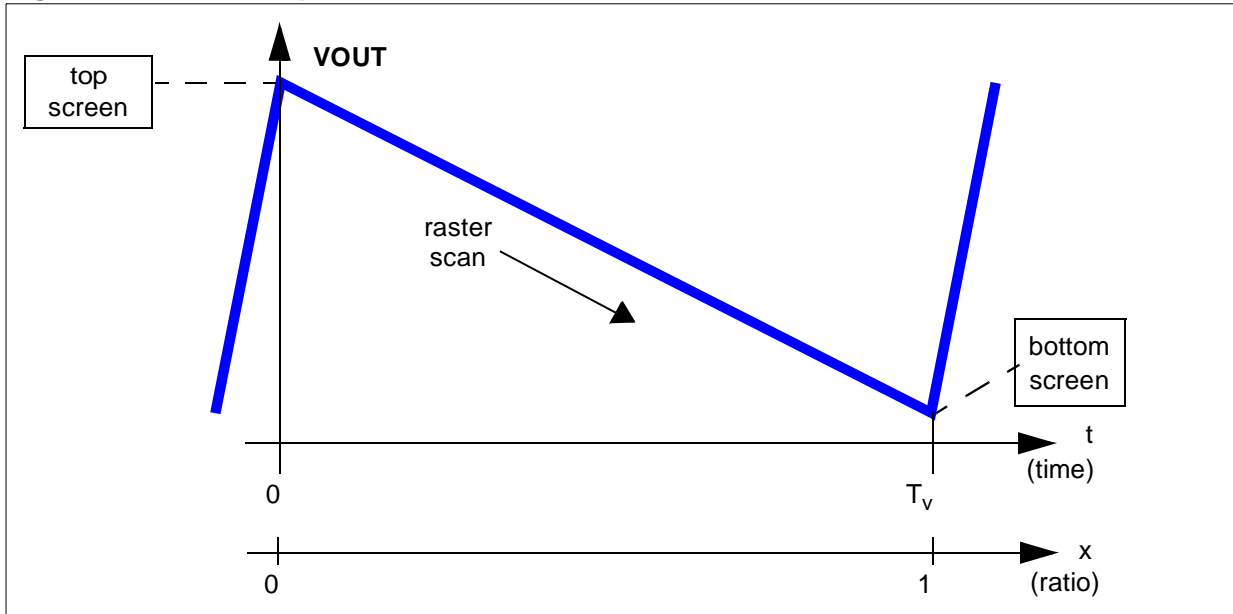
Note: The gain has been *inverted* (“**0.5-x**” instead of “**x-0.5**”) to show the further inversion by the vertical booster driver circuit. This gives a ramp-down curve following the electron beam.

Parameter “*x*” varies linearly as the raster scans the screen, from 0 (top of the screen) to 1 (bottom of the screen). “*t*” is the time elapsed from the top of the screen, “*T*” is the vertical period (T_v) : **the X-axis no more depends on time but always ranges from 0 to 1.**

$$x = \frac{t}{T} \text{ with } x = [0 \dots 1]$$

All computations are simplified and the formula depends on a unique parameter (no unit).

Figure 25. VOUT simplified sawtooth



This formula can be used once a relationship is established between VPOS and VAMP and the corresponding registers values, adjusted by means of I²C commands (as shown in the following extract from TDA 9209 datasheet).

4.5.2 VSIZE setting

VAMP (i.e. VSIZE) is the parameter setting the vertical size of the displayed picture. It is controlled by means of I²C commands (sub-address 05_h).

VSIZE value varies between 00_h (minimum amplitude of 2.25V) and 7F_h (maximum amplitude of 3.75V). Given this, the VAMP (from VOUT formula) derives from VSIZE as follows:

$$V_{AMP} = 2.25 + 1.5 \times \frac{V_{SIZE}}{V_{SIZEMAX}}$$

- n When VSIZE = 0 : VAMP = 2.25V
- n When VSIZE = VSIZEMAX (7F_h) : VAMP = 3.75V

AUTO-SIZING & AUTO-CENTERING

4.5.3 VPOS setting

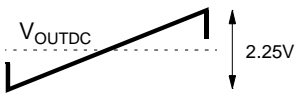

VO_{UTDC} (i.e. VPOS) is the parameter setting the vertical position of the displayed picture. It is controlled by means of I²C commands (sub-address 06_h).

VPOS value varies between 00_h (minimum offset of 3.2V) and 7F_h (maximum offset of 3.8V). Given this, the VO_{UTDC} derives from VPOS as follows :

$$V_{OUTDC} = 3.2 + 0.6 \times \frac{V_{POS}}{V_{POSMAX}}$$

- n When VPOS = 0, VO_{UTDC} = 3.2V
- n When VPOS = VPOS_{MAX} (7F_h), VO_{UTDC} = 3.8V

Figure 26. VSIZE and VPOS settings (TDA 9209)

Function	Sub Address	Pin	Byte	Specification
Vertical Size	05	23	10000000	
			11111111	
Vertical Position DC Control	06	23	x0000000	VO _{UTDC} = 3.2V
			x1000000	VO _{UTDC} = 3.5V
			x1111111	VO _{UTDC} = 3.8V

4.5.4 Complete vertical formula

Considering all previous formulae VO_{UT} is expressed as:

$$V = \left(3.2 + 0.6 \frac{V_{POS}}{V_{POSMAX}} \right) + \left(2.25 + 1.5 \frac{V_{SIZE}}{V_{SIZEMAX}} \right) * \left(0.5 - \frac{t}{T} \right)$$

4.5.5 Reference timing

The above formula only describes the sawtooth output curve after the vertical booster stage. There is no indication about how the picture is to be centered or optimized in size.

For example, there is no guarantee that the picture is centered at $t=T/2$. Actually, the real displayed picture might be shifted along the ramp-down curve where the center of the screen does not necessarily match the center of the curve.

Therefore, the adjustment is used as a **comparison** between a known, well adjusted picture, and a new picture that needs adjusting.

The first picture considered as a **reference**, sets the top and alignment “marks” on the screen: those marks are not identified in space but in **voltage**, as a result from the previous V_{OUT} formula.

As reference for the first picture, we chose the **timing #19** (VGA720X400-7) from pattern generator Chroma 2250:

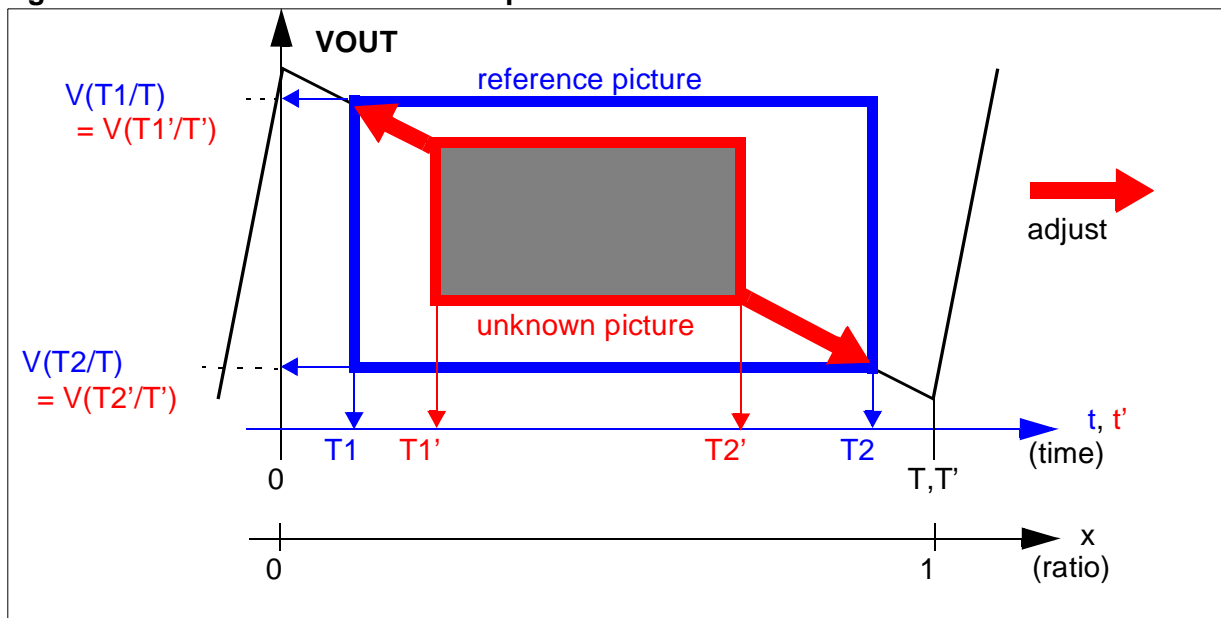
- Horizontal : $f_H = 31.469\text{kHz}$, $T = 31.777\mu\text{s}$
- Vertical : $f_V = 70.087\text{Hz}$, $T = 14.268\text{ms}$, $T1_{AV} = 1.207\text{ms}$, $T2_{AV} = 0.445\text{ms}$, $T_d = 12.616\text{ms}$

This reference timing is used to compute all the constants used in the formulae for HSIZE, VPOS and VSIZE.

The second unknown picture is then adjusted so that its top and bottom lines match the first picture ones in terms of voltage. When those 2 conditions are met, the second picture is assumed to be centered and sized precisely like the first reference picture.

This means stretching the 2nd picture along the vertical sawtooth.

Figure 27. Reference and unknown pictures



T and T' can be different, but x always ranges from 0 to 1.

AUTO-SIZING & AUTO-CENTERING

This leads to two more equations:

TOP OF THE SCREEN: Both picture voltages must be equal after the top black border of the picture. This occurs at $t=T1$ (first reference picture) and $t=T1'$ (second unknown picture), as shown in .

$$V\left(\frac{T1}{T}\right) = V\left(\frac{T1'}{T}\right)$$

BOTTOM OF THE SCREEN: Both picture voltages should be equal before the bottom black border of the picture. This occurs at $t=T2$ (first reference picture) and $t=T2'$ (second unknown picture), as shown in [Figure 6.Vertical Sawtooth on page 9](#).

$$V\left(\frac{T2}{T}\right) = V\left(\frac{T2'}{T}\right)$$

Important Note : T1 and T2 are measured through the TMU cell, by using an AV signal similar to the one shown in [Figure 7.External AV Signal Generated over one Horizontal Line on page 10](#). However, if T1 is measured as shown in figure 7 ($T1=T1_{AV}$), T2 differs from what was previously settled. T2 is the time between the beginning of the screen and the bottom black border of the picture. In the present case, T2 (used in all the equations) becomes $T2 = T - T2_{AV}$ ($T1_{AV}$ and $T2_{AV}$ being returned by the TMU cell).

Solving this 2-equation system leads to the following formulae where the **new VPOS and VSIZE adjustment values** for the unknown second picture are determined, from the first picture reference settings:

NEW VSIZE VALUE:

$$V_{SIZE}' = \frac{1}{2} \cdot (3 \times V_{SIZE_{MAX}} + 2 \times V_{SIZE}) \cdot \left(\frac{Td \times T'}{Td' \times T}\right) - \frac{3}{2} \cdot V_{SIZE_{MAX}}$$

• **Td** is the time of the displayed picture : $Td = (T2 - T1)$ and $Td' = (T2' - T1')$

NEW VPOS VALUE :

$$V_{POS}' = V_{POS} + (a - b)$$

..with :

$$a = \left(2.25 + 1.5 \frac{V_{SIZE}}{V_{SIZE_{MAX}}} \right) \times \left(\frac{1}{2} - \frac{T1}{T} \right) \times \frac{V_{POS_{MAX}}}{0.6}$$

.. and :

$$b = \left(2.25 + 1.5 \frac{V_{SIZE'}}{V_{SIZEMAX}} \right) \times \left(\frac{1}{2} - \frac{T1'}{T} \right) \times \frac{V_{POS MAX}}{0.6}$$

- **VSIZE, VPOS, T1, T2, Td** and **T** all refer to the **first** reference picture.
- **VSIZE', VPOS', T1', T2', Td'** and **T'** all refer to the **second** unknown picture.
- Constant values **VSIZEMAX** and **VPOS MAX** are the maximum reachable value for each respective parameter. For the SAMPO monitor, VSIZEMAX = VPOS MAX = 127.

4.5.6 Operating mode

The computations require a reference picture. The adjustment is performed in several steps :

- First, a given picture is chosen and taken as reference for all future adjustments. This picture is then adjusted carefully, either visually or by any other possible means, so that its centering and sizing fit best with the display area.
 - The corresponding values for VPOS VSIZE T1 T2 and T are then kept as reference and used in all formulas.
- **Note :** It is recommended to store these values as constants (by *#define* or else) within the S/W.
- Next, whenever a new picture is displayed and requires to be adjusted, the new values of T1' T2' and T' are measured through the TMU cell, and VPOS' and VSIZE' values result from the computation of the above parameters, according to the previously established formulae.
- Finally, the deflection processor is updated with those new VPOS' and VSIZE' values: the new picture is nicely adjusted vertically.

Note: In case of overflow: newly computed VSIZE' value exceeding the maximum value that the deflection processor accepts for this setting, the real value and not the rounded one for VSIZE' is used in VPOS' formula. Using the rounded VSIZE' value would lead to a wrong VPOS' value.

4.5.7 Integer formulae

Both VPOS and VSIZE formulae are fairly simple (using the 4 basic operators) but the figures “as is” are of floating type, not integer. Floating numbers are difficult to handle in assembly language. Therefore, a little bit of scaling is needed to facilitate some formulae implementation.

AUTO-SIZING & AUTO-CENTERING

Considering $VSIZE_{MAX} = 127$ (rounded up to 128 with a base-2 number), $VSIZE'$ is rewritten as follows :

$$VSIZE' = \left(\frac{Td \times T'}{Td' \times T} \right) \times (VSIZE + 192) - 192$$

The same method applies for $VPOS'$ with $VPOS_{MAX} = 127$ (rounded up to 128) :

$$VPOS' = VPOS + (a - b)$$

$$a = \frac{5}{4} \times (192 + VSIZE) \times \left(\frac{T - 2T1}{T} \right)$$

$$b = \frac{5}{4} \times (192 + VSIZE') \times \left(\frac{T' - 2T1'}{T'} \right)$$

Every computation is now based on integer numbers (instead of floating previously).

4.5.8 Software implementation

The same remarks about computation as in previous [Section 4.4.8](#) apply here.

Note: Values concerning the reference picture are underlined.

The current implementation for VSIZE is:

- 1 Compute $T' \times \underline{Td}$
- 2 Compute $(T' \times \underline{Td}) / \underline{T}$, then store it
- 3 Compute $((T' \times \underline{Td}) / \underline{T}) \times \underline{VSIZE}$, then store it
- 4 Recall **[2]** then compute $((T' \times \underline{Td}) / \underline{T}) \times 192$
- 5 Add **[3]** plus **[4]** then store it
- 6 Compute $Td' = T' - (T'1+T'2)$, $T'1$ and $T'2$ returned by TMU cell, T' given by syncpro
- 7 Divide **[5]** by **[6]**
- 8 Compute **[7]** - 192: this gives **VSIZE'**
- 9 Compare VSIZE' against maximum setting value (127): if greater, use 127 instead
- 10 Send VSIZE' setting to the deflection processor through I²C commands

The implementation for VPOS is then:

- 11 Compute $T' - 2T'1$
- 12 Compute $192 + \mathbf{[8]}$
- 13 Compute **[10]** x **[11]**
- 14 Compute $5 \times \mathbf{[12]}$
- 15 Divide **[13]** by 4
- 16 Divide **[14]** by T'
- 17 Compute $\underline{CVSIZEREF} - \mathbf{[15]}$: this gives **VPOS'**

Note : $\underline{CVSIZEREF} = \underline{VPOS} + 5/4 \times (192 + \underline{VSIZE}) \times (\underline{T} - 2T'1) / \underline{T}$

- 18 Compare VPOS' against maximum setting value (127): if greater, use 127 instead
- 19 Send VPOS' setting to the deflection processor through I²C commands

5 SET-UP & MEASUREMENTS INSIDE THE MONITOR

The parameters required to implement each separate autoadjustment parameter are now reviewed.

5.1 Horizontal centering (HPOS)

This first algorithm does not require any specific set-up or measurement in the monitor. To help the user understand each possible case as described in [Section 4.3.1.3](#) and [Section 4.3.1.4](#), the following screenshots have been taken on a digital oscilloscope:

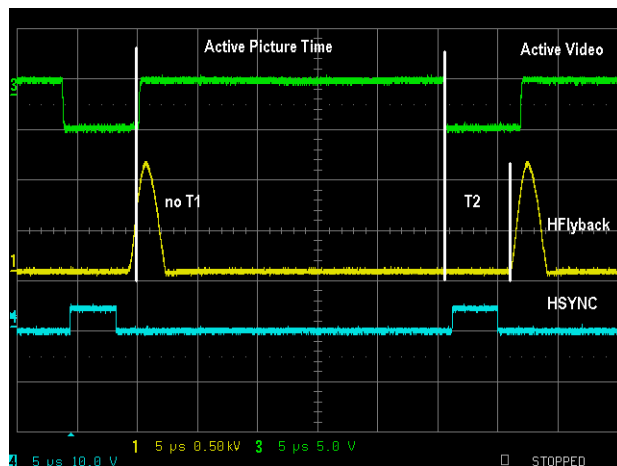
- Top green curve is the Active Video signal, generated by the TDA 9209 pin 8 and fed into the ST72774 pin 2
- Center yellow curve is the HFlyback signal, from the deflection stage (collector pin of transistor Q704 in SAMPO chassis), later sharpened then fed into the ST72774 pin 25
- Bottom blue curve is the positive HSYNC signal, delivered by the ST72774 pin 21

5.1.1 The picture is shifted too far to the left

This extreme case is obtained by setting HPOS = 0 in the monitor menu.

This gives $T1_{AV} < T2_{AV}$.

On the screenshot, the rising edge of AV signal actually occurs much **before** the falling edge of HFlyback : $T1_{AV} = 1$, as described in [Section 3.4](#) and on [Figure 10](#).

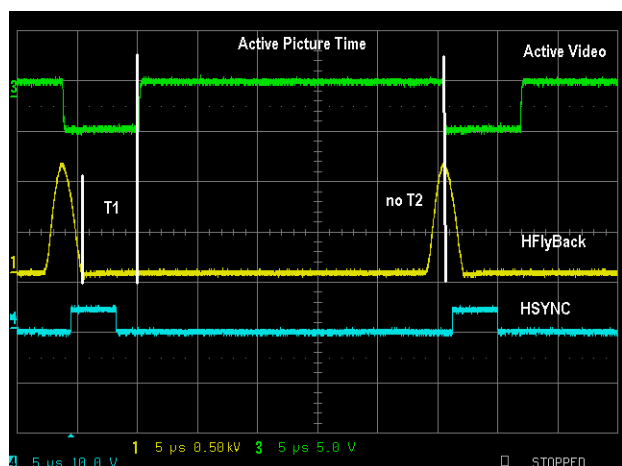


5.1.2 The picture is shifted too far to the right

This extreme case is obtained by setting HPOS = 100 in the monitor menu.

This gives $T1_{AV} > T2_{AV}$.

On the screenshot, the falling edge of AV signal actually occurs much **after** the next rising edge of HFlyback : $T2_{AV} = 1$ (the actual returned value depends on the TMU cell version) as described in [Section 3.4](#) and on [Figure 10](#).



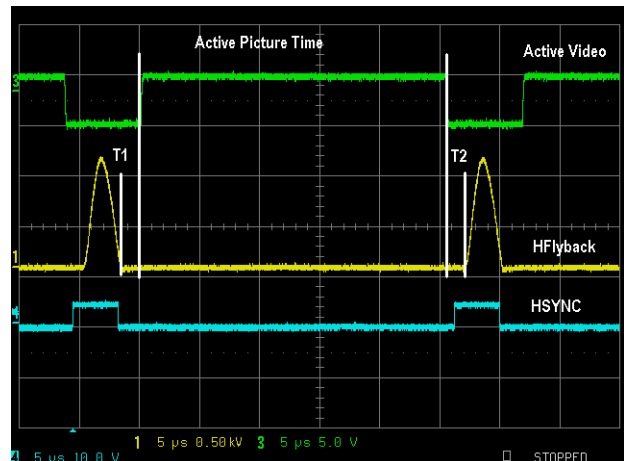
Important Note : As shown on those two screenshots, the HSYNC signal remains in sync with the Active Video signal while HPOS setting changes. Only HFBACK shifts. This is the reason why selecting HSYNC as reference signal for TMU measurements is useless for HPOS algorithm. HFBACK should be used instead.

5.1.3 The picture is perfectly centered

This case is obtained once the horizontal centering algorithm has run (either from [Section 4.3.1](#) or from [Section 4.3.2](#)).

The final value corresponding to the centered HPOS setting strongly depends on which video mode is displayed.

This gives $T1_{AV} = T2_{AV}$.



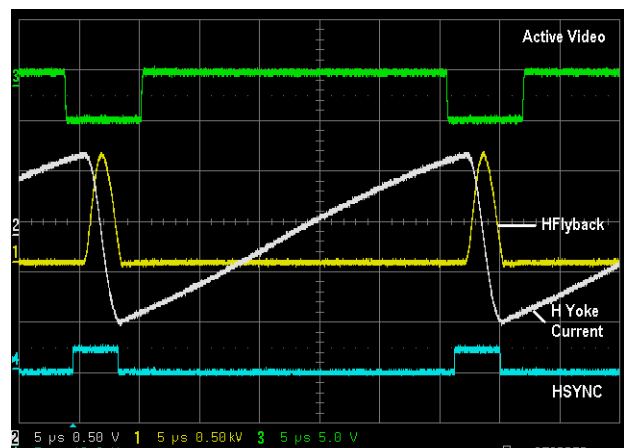
5.2 Horizontal sizing (HSIZE)

This algorithm requires many complex measurements.

To proceed to these measurements, a clip-on current probe (e.g. 10mV/A sensitivity) is clipped onto one of the two wires running from the main PCB to the horizontal deflection yoke. The way the probe is clipped on and which wire is chosen only affects the sign of the curve. It is however advised to get a positive ramp-up current curve on the scope, to match the screenshots below and in [Figure 21](#).

The following screenshots have been taken on a digital oscilloscope with a perfectly centered picture (HPOS already adjusted) :

- n Top green curve is the Active Video signal, generated by the TDA 9209 pin 8 and fed into the ST72774 pin 2
- n First Center yellow curve is the HFlyback signal, from the deflection stage (collector pin of transistor Q704 in SAMPO chassis), later sharpened then fed into the ST72774 pin 25



- n Second Center white curve is the deflection ramp-up current signal going through the horizontal yoke (the S-correction capacitors give this slight “S” shape to the curve)
- n Bottom blue curve is the positive HSYNC signal, delivered by the ST72774 pin 21

5.2.1 Measurement presets

The East-West circuitry (transistor Q719 on SAMPO chassis) controls the horizontal size. Several corrections like pin-cushion, pin balance and trapezoid settings, are applied to this circuitry to fine-tune the horizontal size in a line-by-line basis, and as the raster sweeps the entire display area from top of CRT to bottom.

Due to the above-listed corrections, the overall current amplitude varies from one line to another. Therefore, the settings must be first reset to ensure that the horizontal current curve will remain constant for all the lines across the entire display area.

The following values should be set on the SAMPO monitor via the OSD menu, prior to making any measurement:

- n PINCUSHION = 0
- n PINBALANCE = 50
- n TRAPEZOID = 50

Note: Once set, the values generate unadjusted left and right borders, leading to a strongly distorted picture. This is just temporary, while measurements are performed. The original values must be restored once all measurements have been made.

5.2.2 Current-amplitude measurements

Each min/max measurement is generally made of 5 steps :

- 1 Generate and display the reference video timing as described in [Section 4.5.5](#)
- 2 Preset PINCUSHION, PINBALANCE and TRAPEZOID settings as described above
- 3 Measure the maximum absolute positive value at the top of the HAMP curve, 200ns before HFlyback rises; these 200ns are the real threshold at which the ST72774 input HFBACK senses the high level (VIH) of the HFlyback pulse, in addition to a margin in order to filter out noise and ringing on the signal
- 4 Measure the minimum absolute negative value at the bottom of the HAMP curve, 400ns after HFlyback stops decreasing; these 400ns are the real threshold at which the ST72774 input HFBACK senses the low level (VIL) of the HFlyback pulse, in addition to a margin in order to filter out noise and ringing on the signal
- 5 Compute 3- minus 4- : this gives the absolute overall amplitude of the current curve

5.2.3 Amplitude-measurement of AV_{opti}

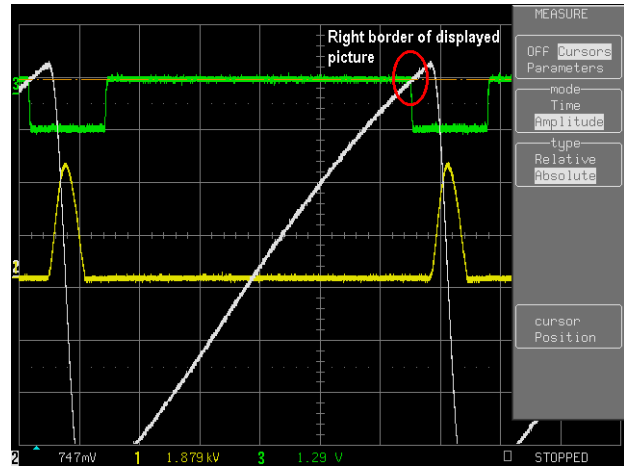
Here is an example of AV_{opti} measurement, AV_{opti} being the constant reference-parameter used to compute all horizontal sizes for all video modes, as described in [Section 4.4.3](#).

The monitor is now assumed to be ready, and Steps 1 and 2 have been conducted. HSIZE setting is manually adjusted until the displayed line at the center of the screen scans the entire

display area from left to right, leaving no black borders. Picture is now assumed to be perfectly fitted both in position and size.

The curve shown on screenshot of [Section 5.2](#) is now zoomed on both horizontal and vertical axis, to focus on the top of the rising edge of the HFlyback signal.

The amplitude of the HAMP signal is then measured when the Active Video signal **falls**, where the red circle is drawn, along the red dotted line : this is the right border of the picture, and the **maximum** value for HAMP.



In this case, the reading is 747mV.

In a similar way, the amplitude of the HAMP signal is then measured when the Active Video signal **rises** : this is the left border of the picture, and the minimum value for HAMP.

In this case, the reading is -773mV.

This gives a final value of $HAMP_{opti} = 747 - (-773) = 1520mV$, hence $A_{Vopti} = HAMP_{opti} / 2.8 = 542$.

Note: This value strongly depends on the current probe used for all measurements, as well as the horizontal deflection circuitry itself. However, the measured value should not change much between similar monitors (same production line model).

5.2.4 Measurements of $HAMP_{MIN}$ and $HAMP_{MAX}$

As explained in [Section 4.4.1](#), two curves are determined for each range of frequencies depending on which S-capacitors are turned off and on :

- $HAMP_{MIN} = f(fH)$ at minimum HSIZE=0
- $HAMP_{MAX} = f(fH)$ at maximum HSIZE=100

In the case of SAMPO monitor, there are 9 ranges of horizontal frequencies but only 7 were fully measured (because as horizontal frequency increases, the width of HFBACK pulse no longer remains constant and has a bigger influence in HSIZE formula). This makes a total of 14 curves to determine, 7 for $HAMP_{MIN}$ and 7 for $HAMP_{MAX}$.

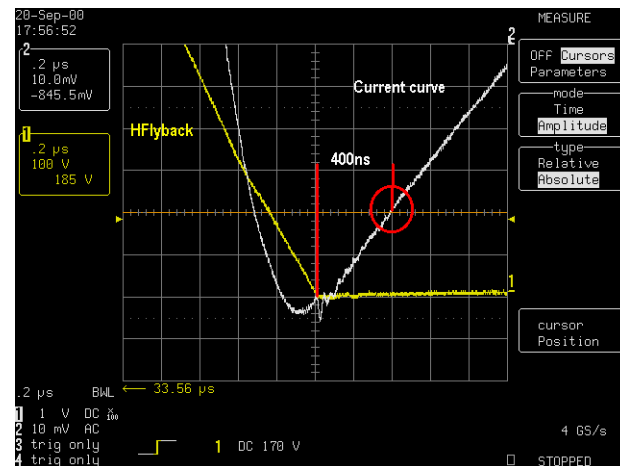
The most precise extrapolation of the equation for each curve is obtained with several measurements of $HAMP_{MIN}$ and $HAMP_{MAX}$ at different frequencies within the same range. A cross chart is established, giving $HAMP_{MIN}$ and $HAMP_{MAX}$ in function of fH. Then a linear extrapolation is performed on all those data, ultimately leading to respectively one formula for $HAMP_{MIN}$ and $HAMP_{MAX}$ at each frequency range.

AUTO-SIZING & AUTO-CENTERING

Each min/max measurement is made exactly the same way as when measuring A_{Vopti} (Section 5.2.3), except that the displayed video mode changes (all valid horizontal frequencies are swept).

For $HAMP_{MIN}$:

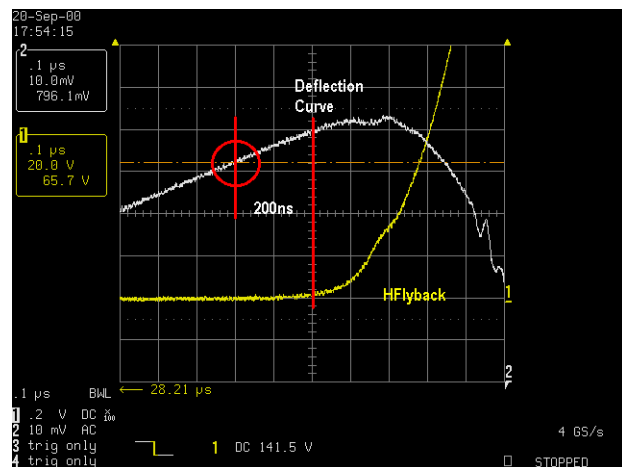
- n Preset PINCUSHION, PINBALANCE and TRAPEZOID settings as described in Section 5.2.1
- n Set HSIZE setting to 0 (smallest picture)
- n Enlarge the picture on the scope to zoom onto the **falling** edge of HFlyback : this is the leftmost border of the display area
- n Read, along the orange dotted line, the value of the current curve amplitude 400ns after the HFlyback curve reaches its minimum (red circle)



In this case, $HAMP_{MIN} = -845mV$

For $HAMP_{MAX}$:

- n Preset PINCUSHION, PINBALANCE and TRAPEZOID settings as described in Section 5.2.1
- n Set HSIZE setting to 100 (largest picture)
- n Enlarge the picture on the scope to zoom onto the **rising** edge of HFlyback : this is the rightmost border of the display area
- n Read, along the orange dotted line, the value of the current curve amplitude 200ns before the HFlyback curve starts to increase (red circle)



In this case, $HAMP_{MAX} = 795mV$

This gives a couple of values (f_H , $HAMP_{MIN}$) and (f_H , $HAMP_{MAX}$) at a given f_H horizontal frequency. The above steps must be repeated for all the valid horizontal frequencies.

Refer to Section 7(ANNEX) for the 14 curves specific to the SAMPO monitor.

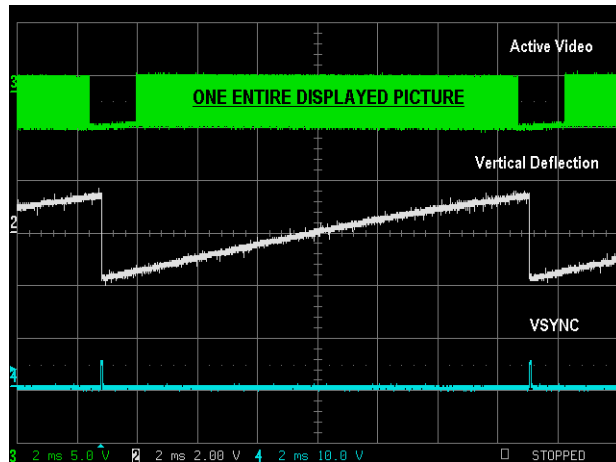
Note : There is no need to take too many values for f_H . A step of 1kHz and/or 5 values per range of frequencies (the extreme 2, and 3 more in between) is sufficient to extrapolate the linear curves.

5.3 Vertical centering/sizing (VPOS/VSIZE)

This algorithm needs no measurements inside the monitor, because reference values for VSIZE and VPOS are used to compute new VSIZE' and VPOS' values for any other video mode.

To help the user understand what is described in [Section 2.3](#) and [Figure 6](#), the following screenshot has been taken on a digital oscilloscope :

- Top green curve is the Active Video signal, generated by the TDA 9209 pin 8 and fed into the ST72774 pin 2; it remains low at the top and the bottom of the picture, where only black remains.



Note : This AV signal is not constant. It is made of several smaller signals as each line is scanned across the screen. There are too many lines to be able to separate them.

- Center white curve is the Vertical deflection signal, from the vertical deflection circuitry (VOUT pin 23 of TDA 9109 chip). The effect of S-correction is notably smaller than on the horizontal deflection curve
- Bottom blue curve is the positive VSYNCO signal, delivered by the ST72774 pin 20

6 END-USER INTERFACE : FINAL TIPS AND FINE-TUNING

6.1 Pictures that work best

As described in [Section 4.2](#), the “RECALL” key up-front has been rerouted to start all the auto-adjustment algorithms previously described.

While the autoadjustment starts, the message “AUTOADJUSTMENT IN PROGRESS” is displayed.

In case of one of the 4 adjustments cannot be performed, a message “AUTOADJUSTMENT FAILED” is displayed for 3 seconds, then normal operation resumes.

Usually, the algorithm(s) fail(s) for various possible reasons previously detailed in this same document.

A main restriction comes from the Active Video generator TDA 9209, as it cannot proceed with video modes which pixel clock is higher than **100MHz** (refer to datasheet, parameter **PixAV** minimum pixel width = 10ns typ.).

This means that any transition between non-black and black levels in a time shorter than **10ns** cannot be sensed, and the AV signal does not change. This is the reason why certain high-resolution pictures, surrounded by a very thin non-black rectangle, will make the autoadjustment procedure fail: outer lines of a few pixels in width may not be sensed properly.

One solution for nearly always successful trials is to display a picture with large non-black borders that delimit sharply the black borders from the real picture. A picture with a full blue background, for example, gives excellent results.

6.2 End-user fine-tuning

All these algorithms involve that constant values are measured and/or computed from a reference video mode, generated by a pattern generator (refer to [Section 4.5.5](#)). These constants are:

- n A_{Vopti}
- n VPOS
- n VSIZE
- n CVSIZEREF (computed from VPOS and VSIZE above)

In practice, these constants are slightly adjusted between monitor displays of the same model, due to uneven factory adjustments (such as spot beam centering). For example, one screen displayed a perfectly centered picture when another displayed a slightly shifted picture.

To avoid customizing these constants inside the S/W for each screen (which is tedious and definitely not convenient), the decision was made to implement a new OSD menu to allow the user to :

- freely adjust $A_{V_{opti}}$, VPOS and VSIZE
- recalculate CVSIZEREF from VPOS and VSIZE above

Important Note : For these adjustments, the reference mode #19 (refer to [Section 4.5.5](#)) must be displayed as all computations are based on it.

The rerouted OSD-menu-feature is the former “RECALL” icon, one step below the “EXIT” icon (this is 1 anticlockwise-step of the rotary knob). It displays “SETUP TMU”, then, when clicking on the rotary knob, brings an autoadjustment-dedicated submenu.



A second click on the rotary knob leaves this submenu and stores the adjusted parameters into the monitor **EEPROM** memory for later use.

Caution : This adjustment must be done once, at least, and always after the very first power-on because the existing SAMPO S/W defaults the entire EEPROM contents to 00h.

6.2.1 Optimum HAMP size

In the OSD submenu, the user adjusts HAMP opti ($= A_{Vopti} / 2.8$) freely between 000h and FFFh in **hexadecimal** value, by rotating the knob (step **7-** in [Section 4.5.8](#)).

This parameter is directly proportional to the final HSIZE of the adjusted displayed picture: the smaller the value, the smaller the picture. As a reminder, the original value is $A_{Vopti} = 1130$ (refer to [Section 4.4.3](#)), hence $HAMP\ opti = A_{Vopti} / 2.8 = \mathbf{193h}$. The value is directly measured in the monitor as described in [Section 5.2.3](#).

By slightly adjusting HAMP opti, the best fitted picture in horizontal size is obtained.

Note : The new parameter value is valid only for the next autoadjustment routines. It does not change the present horizontal size “visually” (while being modified).

Caution : It is not recommended, although possible, to deviate too far from the nominal measured value, as the resulting computed HSIZE will not be quite as expected.

6.2.2 Optimum VPOS and VSIZE

As previously described for HAMP, VPOS and VSIZE reference values are used to give the best fitted picture in both vertical centering and sizing.

If the adjusted picture deviates from its optimum center position and/or vertical size, new VPOS and VSIZE reference parameters are stored:

- 1 Display the reference video mode #19 (for Chroma processor series)
- 2 Perform the autoadjustment once by pressing the “RECALL” key upfront
- 3 Bring the main OSD menu by clicking on the rotary knob
- 4 Select V-POSITION icon (the rightmost on the upper line)
- 5 Adjust the vertical position to best center the picture
- 6 Select V-SIZE icon (the top first on the rightmost line)
- 7 Adjust the vertical size to best fit the picture with the screen
- 8 Select the former RECALL icon, to bring the SETUP TMU submenu (see [Section 6.2.1](#)). Just by entering the SETUP TMU submenu, both VPOS and VSIZE values are stored into the EEPROM.
- 9 Exit this submenu

The new parameter values are valid effect only for the next autoadjustment routines. Further autoadjustments also recalculate the CVSIZEREF constant parameter, using the above values stored as reference.

7 ANNEX

The 14 curves pertaining to the 7 ranges of horizontal frequencies for the SAMPO monitor are now provided. Each curves shows on its 2 axis :

- X axis : horizontal frequency fH (x1kHz)
- Y axis : HAMP opti (x10mV/A)

Each green dot is a couple (fH, HAMP opti) within the same range of horizontal frequencies

The resulting linear equation expresses HAMP opti in function of fH.

The provided linear regression factor R is always very close to 1, which means that the linear extrapolation is near perfect (very close to a straight line) except for range fH between 31kHz and 34kHz due to the too few measurements available (it was not so easy on the Chroma processor to generate video modes with horizontal frequency sub-step of 0.5kHz instead of 1kHz)

For each range of horizontal frequencies, the top curve is for HAMP_{MIN} (HSIZE=0), and the bottom curve is for HAMP_{MAX} (HSIZE=100).

Figure 28. HAMP opti = f(fH) for fH between 31kHz and 34kHz

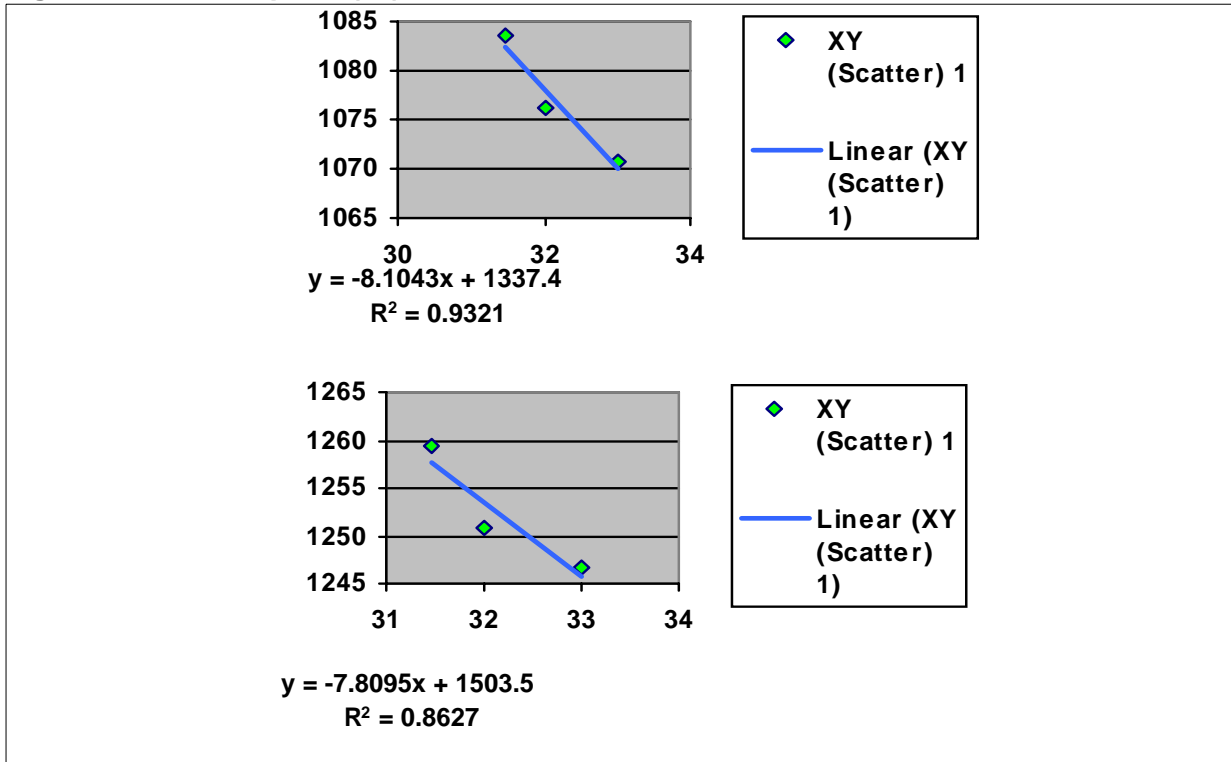


Figure 29. HAMP opti = f(fH) for fH between 34 and 40.6kHz

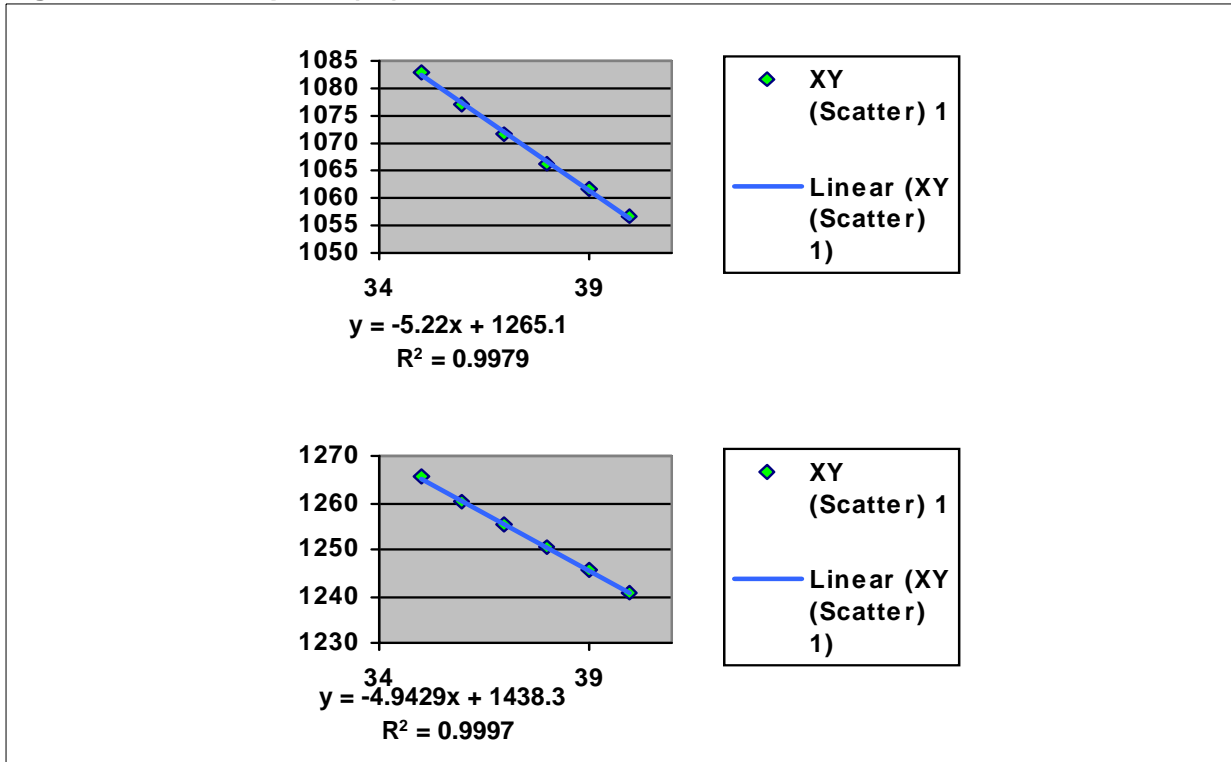


Figure 30. HAMP opti = f(fH) for fH between 40.6kHz and 44kHz

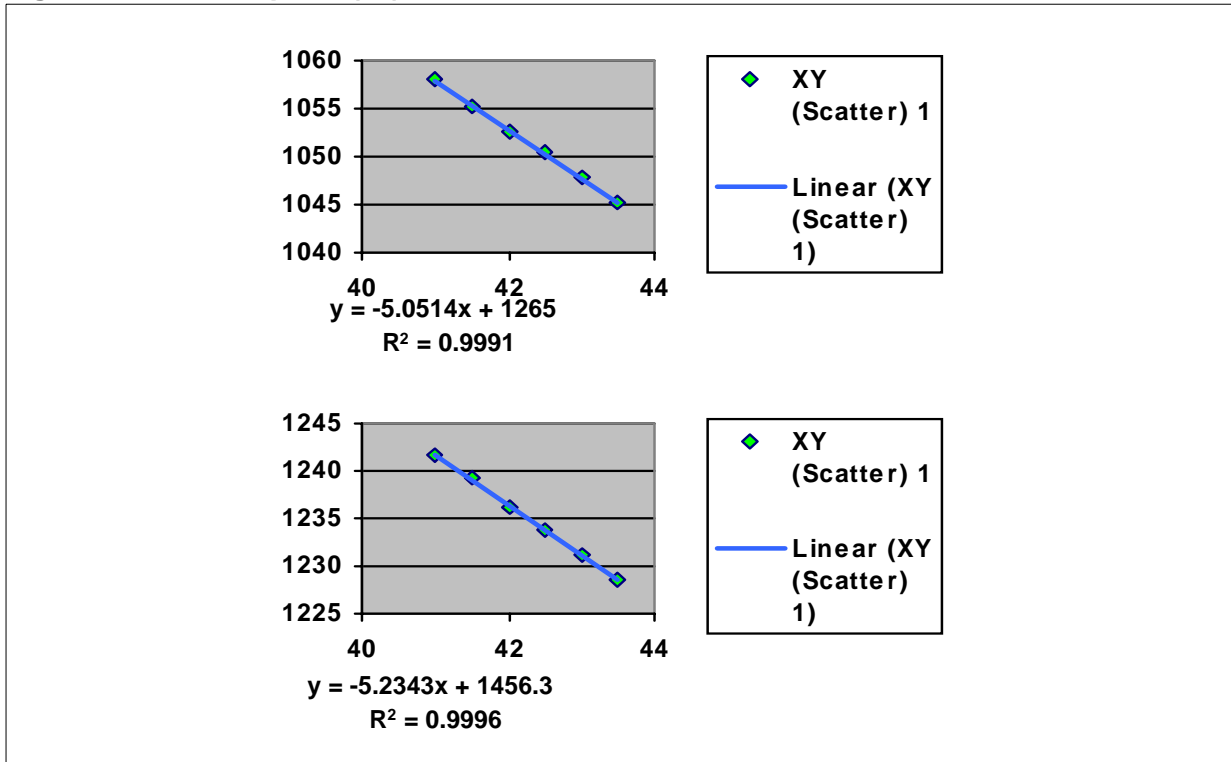


Figure 31. HAMP opti = f(fH) for fH between 44kHz and 59.25kHz

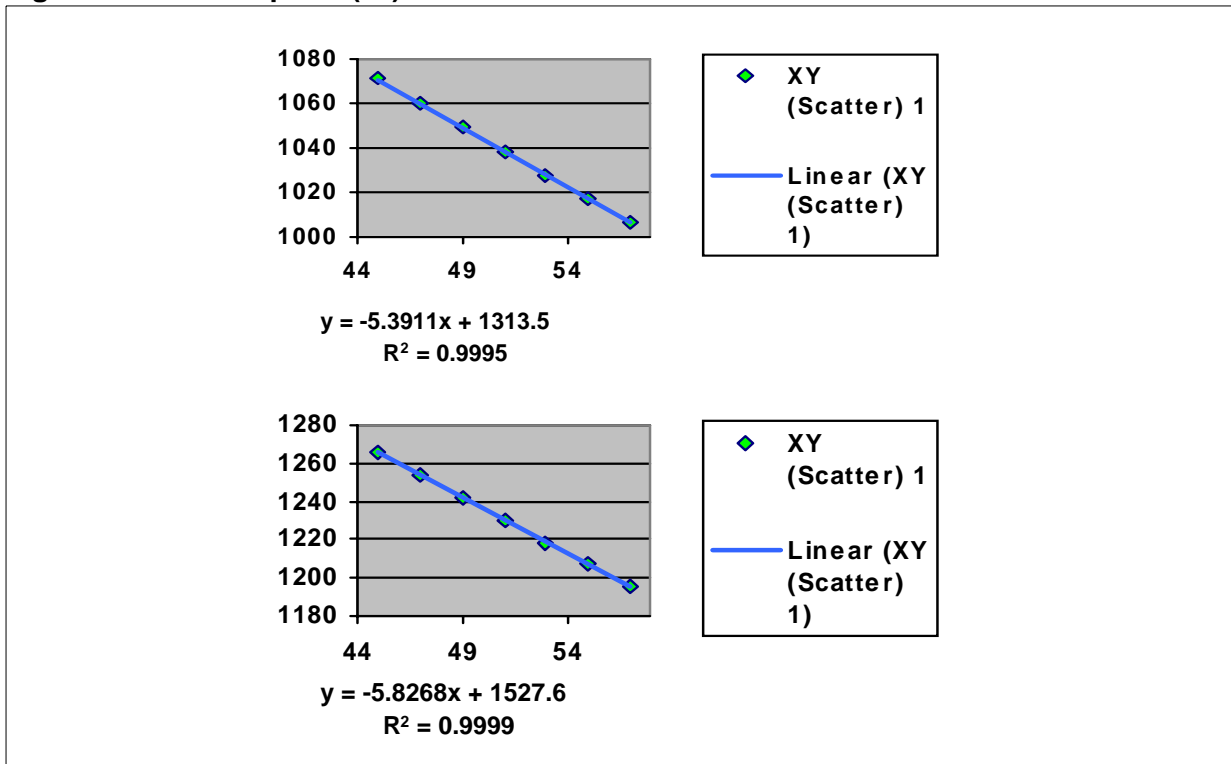


Figure 32. HAMP opti = f(fh) for fh between 59.25kHz and 68kHz

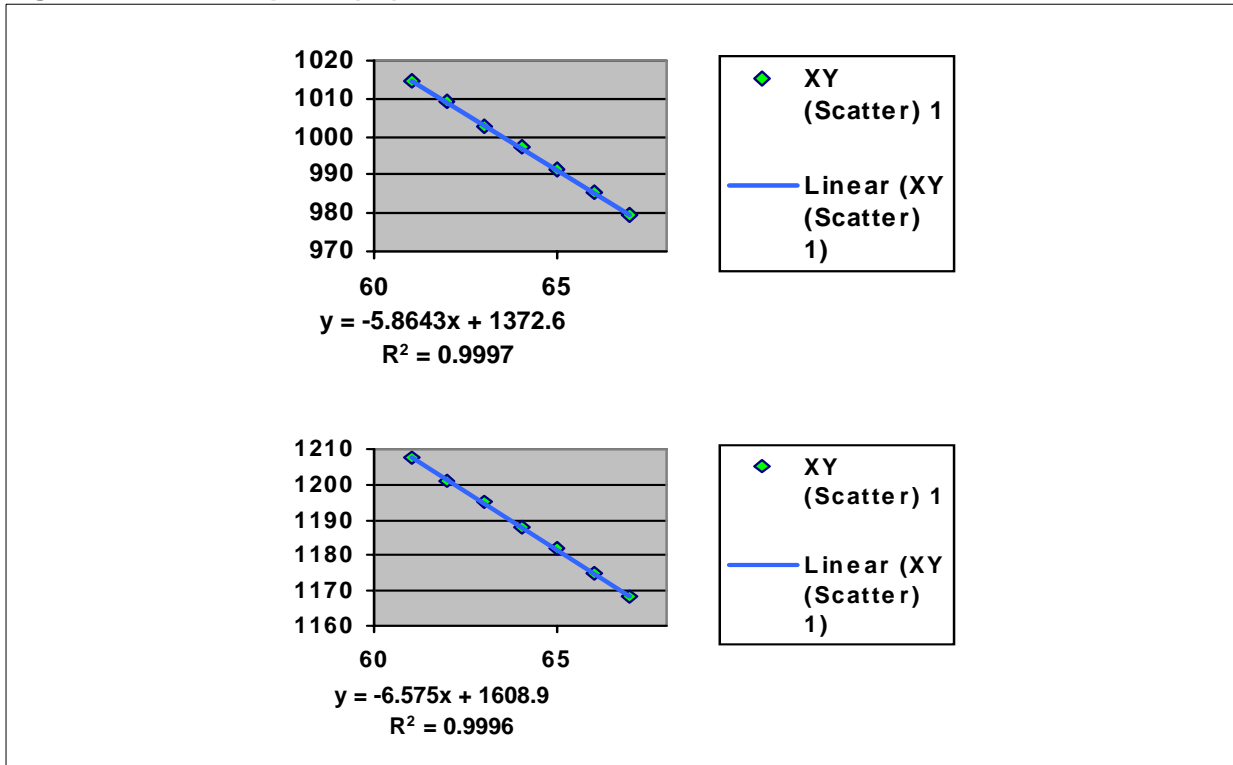


Figure 33. HAMP opti = f(fh) for fh between 68kHz and 73.25kHz

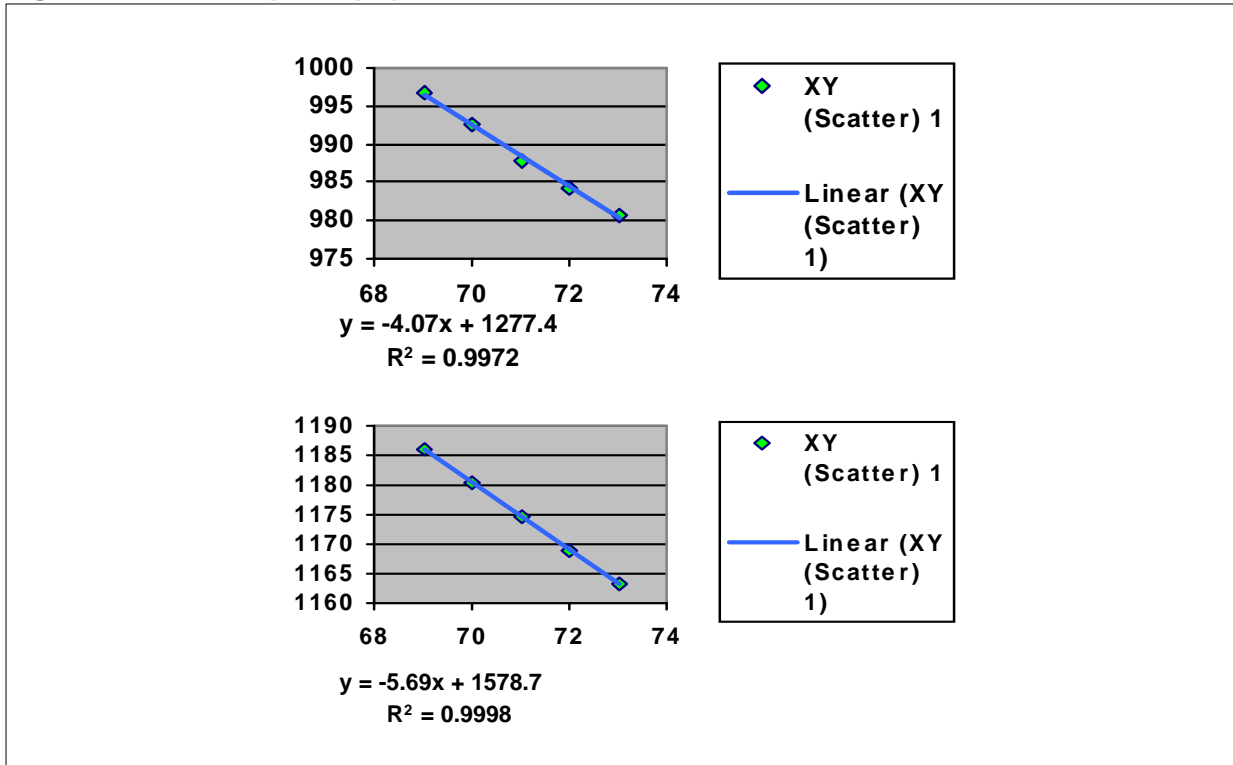
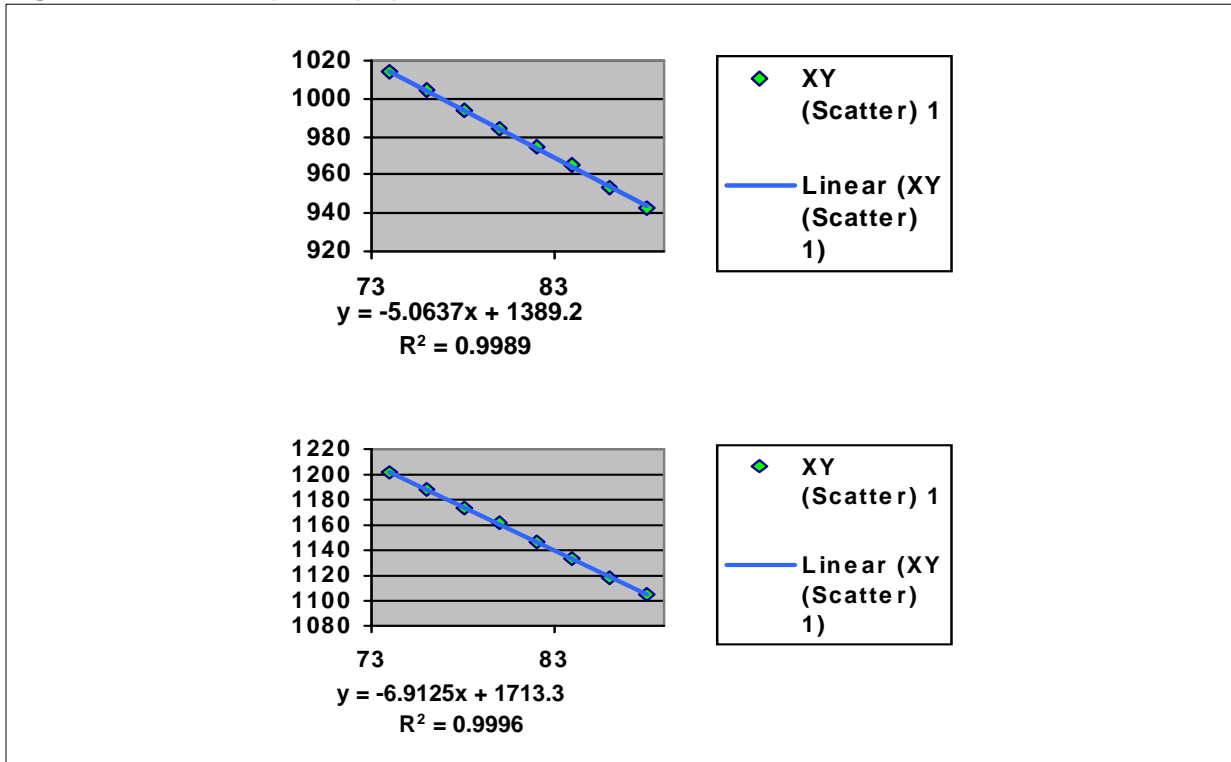


Figure 34. HAMP opti = f(fH) for fH between 73.25kHz and 89.25kHz



Note : As written in [Section 5.2.4](#), the last 2 ranges of horizontal frequencies (89..95kHz and beyond 95kHz) were not measured.

"THE PRESENT NOTE WHICH IS FOR GUIDANCE ONLY AIMS AT PROVIDING CUSTOMERS WITH INFORMATION REGARDING THEIR PRODUCTS IN ORDER FOR THEM TO SAVE TIME. AS A RESULT, STMICROELECTRONICS SHALL NOT BE HELD LIABLE FOR ANY DIRECT, INDIRECT OR CONSEQUENTIAL DAMAGES WITH RESPECT TO ANY CLAIMS ARISING FROM THE CONTENT OF SUCH A NOTE AND/OR THE USE MADE BY CUSTOMERS OF THE INFORMATION CONTAINED HEREIN IN CONNEXION WITH THEIR PRODUCTS."

Information furnished is believed to be accurate and reliable. However, STMicroelectronics assumes no responsibility for the consequences of use of such information nor for any infringement of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of STMicroelectronics. Specifications mentioned in this publication are subject to change without notice. This publication supersedes and replaces all information previously supplied. STMicroelectronics products are not authorized for use as critical components in life support devices or systems without express written approval of STMicroelectronics.

The ST logo is a trademark of STMicroelectronics.

© 2001 STMicroelectronics - All Rights Reserved

Purchase of I2C Components of STMicroelectronics, conveys a license under the Philips I2C Patent. Rights to use these components in a I2C system, is granted provided that the system conforms to the I2C Standard Specifications as defined by Philips.

STMicroelectronics GROUP OF COMPANIES

Australia - Brazil - China - Finland - France - Germany - Hong Kong - India - Italy - Japan - Malaysia - Malta - Morocco - Singapore - Spain - Sweden - Switzerland - United Kingdom - U.S.A.

<http://www.st.com>