How to Use DSO138 Library

Applicable library version: 13803-031 or newer

1. DSO138 Capture Engine Model

Fig. 1 shows the structure of the DSO138 capture engine.

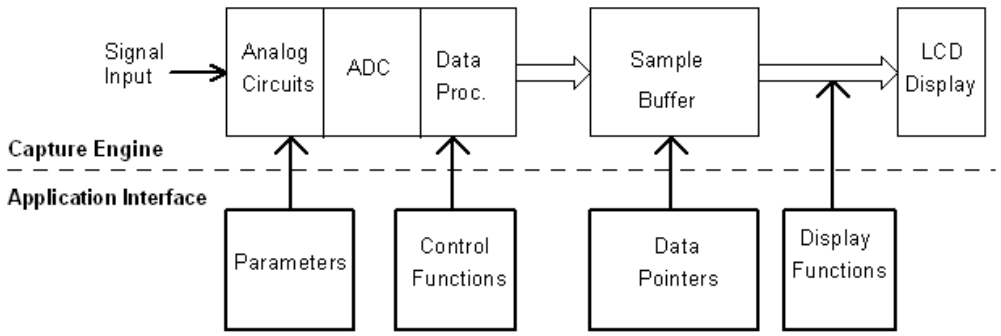


Fig. 1

Input signals are first conditioned by the analogue circuits. They are then fed to analogue-to-digital converter (ADC) which samples and converts them to digital values (also called as samples). These digital values are stored to a specific memory (Sample Buffer) after certain processing and displayed on LCD as waveforms with the help of display functions.

The capture process is governed by a set of parameters including sensitivity, couple, time base, etc. and can be manipulated by a group of control functions. The stored data are accessible by means of data pointers. The parameters, control functions, data pointers, and display functions constitute the application interface of capture engine.

2. DSO138 Library

DSO138 library functions are categorized into three major groups, Parameter Access Functions, Capture Control Functions, and Display Functions. The library also provides two data pointers and a function for accesses to the sample buffer. These functions and pointers are explained below one by one.

1) Parameter Access Functions

Accessible parameters of DSO138 include:

- Sensitivity (VSen)
- Couple (Cpl)
- Vertical Position (VPos)
- Time base (TimeBase)
- Horizontal Position (HPos)
- Trigger Mode (TrigMode)
- Trigger Slope (TrigEdge)
- Trigger Level (TrigLvl)

- Record Length (RecLen)
- Vertical Position Offset (VPosOfs)

Syntax	**S8 SetVSen(S8 *vsen*);**
Parameters	*vsen* – the new sensitivity to be set. Sensitivity can be one of the following values: VS_5V (0x04) VS_2V (0x05) VS_1V (0x06) VS_05V (0x07) VS_02V (0x08) VS_01V (0x09) VS_50mV (0x0A) VS_20mV (0x0B) VS_10mV (0x0C)
Return value	The current sensitivity setting
Remarks	For DSO138 this function only changes the internal value of the sensitivity setting and has no effect on actual circuit because actual sensitivity is completely dependant on SW2 and SW3. Internal sensitivity setting will be over-written by actual value whenever SW2 or SW3 is moved.

Syntax	**S8 GetVSen(void);**
Parameters	None
Return value	The current sensitivity internal value
Remarks	Returns the current sensitivity internal value.

Syntax	**S8 SetCpl(S8 *cpl*);**
Parameters	*cpl* – the new couple to be set. Couple can be one of the following values: CP_DC (0x00) CP_AC (0x01) CP_GND (0x02)
Return value	The current couple setting
Remarks	For DSO138 this function only changes the internal value of the couple setting and has no effect on actual circuit because actual couple is completely dependant on switches SW1. Internal setting will be over-written by actual value whenever SW1 is moved.

Syntax	**S8 GetCpl(void);**
Parameters	None
Return value	The current couple internal setting
Remarks	Returns the current couple internal setting.

Syntax	**S16 SetVPos(S16 *vpos*);**
Parameters	*vpos* – the new vertical position to be set. The value range of vertical position is -255 – 255 with 0 corresponding to the vertical centre of waveform window.
Return value	The current vertical position value
Remarks	This function sets the vertical position of waveform display.

Syntax	**S16 GetVPos(void);**
Parameters	None
Return value	The current vertical position value
Remarks	This function returns the current vertical position of waveform display.

Syntax	**S8 SetTimeBase(S8 *timebase*);**
Parameters	*timebase* – the new time base to be set. Time base can be one of the following values (numbers in square brackets are corresponding sampling rates): TB_500s (0x01) [0.05Hz (20s)] TB_200s (0x02) [0.125Hz (8s)] TB_100s (0x03) [0.25Hz (4s)] TB_50s (0x04) [0.5Hz (2s)] TB_20s (0x05) [1.25Hz (0.8s)] TB_10s (0x06) [2.5Hz (0.4s)] TB_5s, (0x07) [5Hz (0.2s)] TB_2s (0x08) [12.5Hz] TB_1s (0x09) [25Hz] TB_05s (0x0A) [50Hz] TB_02s (0x0B) [125Hz] TB_01s (0x0C) [250Hz] TB_50ms (0x0D) [500Hz] TB_20ms (0x0E) [1.25KHz] TB_10ms (0x0F) [2.5KHz] TB_5ms (0x10) [5KHz] TB_2ms (0x11) [12.5KHz] TB_1ms (0x12) [25KHz] TB_05ms (0x13) [50KHz] TB_02ms (0x14) [125KHz] TB_01ms (0x15) [250KHz] TB_50us (0x16) [500KHz] TB_20us, (0x17) [1.25MHz*] TB_10us (0x18) [2.5MHz*] * These are equivalent sampling rate.
Return value	The current time base setting
Remarks	This function sets the time base setting.

Syntax	**S8 GetTimebase(void);**
Parameters	None
Return value	The current time base setting
Remarks	This function returns the current time base setting.

Syntax	**S16 SetHPos(S16 *hpos*);**
Parameters	*hpos* – the new horizontal position to be set. The value range of horizontal position is from 0 to (Record Length – 300) with 0 corresponding to the left-most position.
Return value	The current horizontal position value
Remarks	This function sets the horizontal position of waveform display.

Syntax	**S16 GetHPos(void);**
Parameters	None

Return value	The current horizontal position value
Remarks	This function returns the current horizontal position of waveform display.

Syntax	**S8 SetTrigMode(S8 *trigmode*);**
Parameters	*trigmode* – the new trigger mode to be set. Trigger mode can be one of the following values: TM_Auto (0x00) TM_Normal (0x01) TM_Single (0x02)
Return value	The current trigger mode setting
Remarks	This function sets the trigger mode.

Syntax	**S8 GetTrigMode(void);**
Parameters	None
Return value	The current trigger mode
Remarks	This function returns the current trigger mode setting

Syntax	**S8 SetTrigEdge(S8 *trigslope*);**
Parameters	*trigslop* – the new trigger slope to be set. Trigger slope can be one of the following values: TE_Falling (0x00) TE_Rising (0x01)
Return value	The current trigger slope setting
Remarks	This function sets the trigger slope.

Syntax	**S8 GetTrigEdge(void);**
Parameters	None
Return value	The current trigger slope
Remarks	This function returns the current trigger slope setting

Syntax	**S16 SetTrigLvl(S16 *triglvl*);**
Parameters	*triglvl* – the new trigger level to be set. The value range of trigger level is from -300 to 300 with 0 corresponding to 0V level.
Return value	The current trigger level
Remarks	This function sets the trigger level.

Syntax	**S16 GetTrigLvl(void);**
Parameters	None
Return value	The current trigger level
Remarks	This function returns the current trigger level.

Syntax	**U16 SetRecLen(U16 *reclen*);**
Parameters	*reclen* – the new record length to be set. The value range of record length is from 512 to 1024.
Return value	The current record length
Remarks	This function sets the record length.

Syntax	**U16 GetRecLen(void);**
Parameters	None
Return value	The current record length

Remarks	This function returns the current record length.

Syntax	**S16 SetVPosOfs(S16 *ofs*);**
Parameters	*ofs* – the new vertical position offset. The value range of trigger level is from -255 to 255.
Return value	The current vertical position offset
Remarks	This function sets the vertical position offset. Vertical position offset is used to correct the possible mismatch between 0V trace and vertical position indicator.

Syntax	**S16 GetVPosOfs(void);**
Parameters	None
Return value	The current vertical position offset
Remarks	This function returns the current vertical position offset.

Note: In most cases newly changed parameters will not take effect until function StartCapture() (see below) is executed.

2) <u>Capture Control Functions</u>

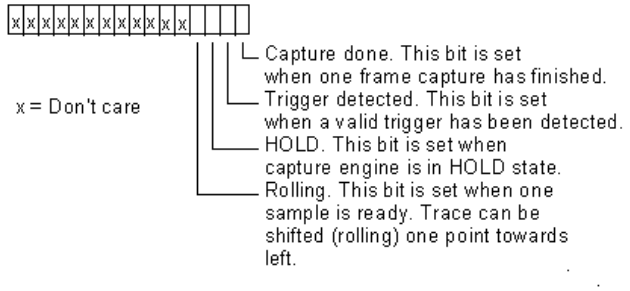
The capture engine runs in two modes based on time base setting. When time base is set to faster than 50ms/div (i.e. 20ms/div or faster) the capture engine runs in **Frame Mode**, which means waveform display will only be updated after the whole sample buffer has been filled up with samples. When time base is set to 50ms/div or slower capture engine runs in **Rolling Mode**. In this mode waveform trace will be shifted horizontally from right to left one point at each new sample.

Syntax	**void DSO_Init(void);**
Parameters	None
Return value	None
Remarks	This function performs the necessary initialization for the capture engine.

Syntax	**void StartCapture(void);**
Parameters	None
Return value	None
Remarks	This function starts a new capture with the current parameters. Any undergoing capture will be aborted.

Syntax	**void StopCapture(void);**
Parameters	None
Return value	None
Remarks	This function stalls capture engine until next StartCaptuer() is executed.

Syntax	**U16 GetDsoStatus(void);**
Parameters	None
Return value	The current capture engine status. The meaning of capture engine status:

	
Remarks	This function returns the current status of capture engine. The status bits are read-only. All status bits except the bit for HOLD are cleared when function StartCapture() is executed.

Syntax	**void SetHold(void);**
Parameters	None
Return value	None
Remarks	This function puts the capture engine into HOLD state. Waveform refresh will be frozen until ClrHold() is performed.

Syntax	**void ClrHold(void);**
Parameters	None
Return value	None
Remarks	This function releases capture engine from HOLD state and restores capture.

Syntax	**void UpdateTimebase(void);**
Parameters	None
Return value	None
Remarks	This function puts time base (which is usually newly changed) in effect immediately instead of waiting for the execution of StartCapture() function.

3) Display Functions

Syntax	**void DsoDisplay(void);**
Parameters	None
Return value	None
Remarks	This function responds to the display requests (sent by function UpdateDisp() and renders scope panel (grids), parameters, and traces accordingly.

Syntax	**void UpdateDisp(U16 *disp*);**
Parameters	*disp* – display request Display request can be one of the following values or their combination. Disp_Panel (0x0001) – Update panel display Disp_Param (0x0002) – Update parameter display Disp_Trace (0x0004) – Update trace display Disp_None (0x0008) – Clear screen
Return value	None

Remarks	This function sends various display requests to function DsoDisplay().

Syntax	**void Rolling(void);**
Parameters	None
Return value	None
Remarks	This function shifts waveform trace towards left by one point (one sample).

Syntax	**U8 SetFocus(U8 *focus*);**
Parameters	*focus* – item to be highlighted (focused) Focus can be one of the following values: FC_Timebase (0x00) – Timebase FC_TrigMode (0x01) – Trigger Mode FC_TrigEdge (0x02) – Trigger Slope FC_TrigLvl (0x03) – Trigger Level FC_HPos (0x04) – Horizontal Position FC_VPos (0x05) – Vertical Position
Return value	None
Remarks	This function is used to select and highlight the specified parameter so as it can be adjusted.

Syntax	**U8 GetFocus(void);**
Parameters	None
Return value	The currently highlighted (focused) parameter
Remarks	This function returns the currently highlighted parameter.

4) Data Access Pointers and Functions

Captured data are stored in a segment of specific memory (Sample Buffer). Each sample is half-word (16bits) in size. But only the lower 12 bits are significant. The highest 4 bits are always zeros. The size of Sample Buffer (in half-word) is equal to the record length setting. To access the Sample Buffer two pointers and one function can be used.

Definition	**U16 ****SampleBuf***;**
Remarks	This is a pointer that always points to the start of sample buffer. You can use this pointer to access captured data when the status bit of CaptureDone is set.

Definition	**U16 ****CurrentSample***;**
Remarks	This is a pointer that points to the last sample acquired. Please note that this pointer is only valid when capture engine is running in Rolling Mode. In Frame Mode *CurrentSample* does not contain a defined value.

Syntax	**S16 GetAverage(void);**
Parameters	None
Return value	The average of sample values in Sample Buffer
Remarks	This function returns the average value of all samples that have been stored in the Sample Buffer.

3. How to Use the Library

1) Conditions must be met for the capture engine to function

In order to have the capture engine function properly the following conditions must be satisfied.

a) The function DSO_Init() must be called prior to execution of function NVIC_Configuration() and entering the main loop.

b) NVIC_Configuration() must be called before entering the main loop. Do not modify the lines for TIM1, ADC1, and DMA Channel 1 interrupt settings inside function NVIC_Configuration().

c) The function DsoDisplay() must be kept in the main loop even if you don't want the scope panel be shown.

d) The following lines much be included in the main loop even if you don't use any scope functions.

```
if(GTimeout) {
    GTimeout = 0;
    StartCapture();
    }
```

e) Do not use TIM1, TIM2, ADC1, and DMA Channel1 or change their settings. These peripherals are exclusively used by the capture engine.

2) Build your application with the library

The library is provided with two files, libdso138.a and libdso138.h. They are developed under Sourcery CodeBench Lite from Mentor Graphics. To build it into your application just include the header file libdso138.h into your source codes and link them with libdso138.a. Please take the DSO138 source code package an example.

3) Add features to the oscilloscope

You can use the DSO138 source code package as a base and add other features you like on top of it. Just keep the conditions met as stated in 1) and you are free to modify other codes.

4) Develop applications without the oscilloscope

If you want to develop your application using the capture engine but don't want the oscilloscope panel displayed you just need to set the variable NoScopePanel to non-zero value before entering the main loop and do not send any display requests with function UpdateDisp(). The scope panel will not show and you are free to add your own codes. Remember, you still need to make sure the conditions listed in 1) above are met.

Revision History

Revision	Date	Summary
v01	2014.12.25	Draft
v02	2014.12.31	Added descriptions for sampling rates in the function SetTimeBase().