# DM8168 DVR RDK Application and GUI Guide


DVR RDK Version: 02.00.00.xx

## IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

| **Products** | | **Applications** | |
|---|---|---|---|
| Amplifiers | amplifier.ti.com | Audio | www.ti.com/audio |
| Data Converters | dataconverter.ti.com | Automotive | www.ti.com/automotive |
| DSP | dsp.ti.com | Broadband | www.ti.com/broadband |
| Clocks and Timers | www.ti.com/clocks | Digital Control | www.ti.com/digitalcontrol |
| Interface | interface.ti.com | Medical | www.ti.com/medical |
| Logic | logic.ti.com | Military | www.ti.com/military |
| Power Mgmt | power.ti.com | Optical Networking | |
| | www.ti.com/opticalnetwork | | |
| Microcontrollers | microcontroller.ti.com | Security | www.ti.com/security |
| RFID | www.ti-rfid.com | Telephony | www.ti.com/telephony |
| RF/IF and ZigBee® Solutions | www.ti.com/lprf | Video & Imaging | www.ti.com/video |
| | | Wireless | www.ti.com/wireless |

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas,
Texas 75265 

## *Revision History*

| Version | Date | Revision History |
|---|---|---|
| 0.1 | Sept 29 , 2011 | GUI Demo Guide for DVR RDK release version v01.05 |
| 0.2 | Nov 5 , 2011 | GUI Demo Guide for DVR RDK release version v01.06 |
| 0.3 | Dec 30 , 2011 | GUI Demo Guide for DVR RDK release version v01.08 |
| 0.4 | Feb 1, 2012 | GUI Demo Guide for DVR RDK release version v01.09 |
| 0.5 | Feb 28, 2012 | GUI Demo Guide for DVR RDK release version v02.00 |

# Table of Contents

# 1. Introduction

The document covers the details for DVR Application and GUI used in DM8168 DVRRDK. Please refer to DM8168_DVR_RDK_InstallGuide.pdf and DVR_RDK_McFW_UserGuide.pdf for DVR RDK installation steps.

# 2. Building the DVR Application and GUI

Assuming the application is already installed in <DVR_RDK_INSTALL_DIR>, the following sections provide details on how to build the DVR application.

**Note: DVR GUI and Application has been validated to work on DVR RDK Hardware**

## 2.1 Building Application

- Move to base directory of "dvr_rdk"

```
$ cd <DVR_RDK_INSTALL_DIR>/dvr_rdk
```

- Build application using option.

```
$ make dvrapp
```

- After the above step for building, the binaries are available at `<DVR_RDK_INSTALL_DIR>/dvr_rdk/bin/ti816x/bin`. The binaries are automatically copied to the following directory.
  - ✓ libdm816x.so to `<DVR_RDK_INSTALL_DIR>/target/rfs/opt/dvr_rdk/ti816x/firmware/`
  - ✓ dvrmain to `<DVR_RDK_INSTALL_DIR>/target/rfs/opt/dvr_rdk/ti816x/bin`
- To copy the binaries to the root file system, use the following commands

```
$ make fsupdate
```

## 2.2 Building GUI

- Refer to Qt_Installation_Guide.pdf for setting up Qt
- Edit `<DVR_RDK_INSTALL_DIR>/dvr_rdk/dvrapp/dvrgui/qmake.sh` with your own directory path
- Build GUI using following command inside `<DVR_RDK_INSTALL_DIR>/dvr_rdk` folder

```
$ make dvrgui
```

- To copy the binaries to the root file system, use the following commands

```
$ make fsupdate
```

# 3. Executing the GUI Application

- When booting is finished, user can login as "root" on the serial terminal. **Note that password is not required.**

```
dvr login: root
```

- Move to target directory and execute the shell script "start_app.sh" as below

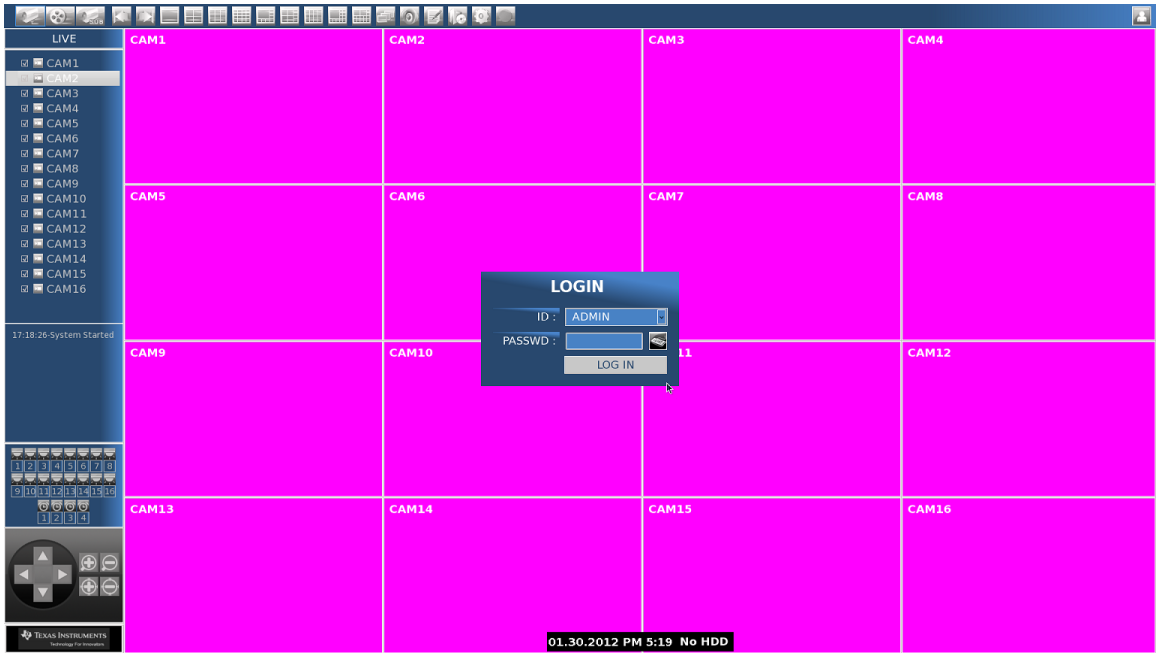
```
#!/bin/sh

# start application ----------------------------
echo -e "start application"
app_dir="dvr_rdk/ti816x"

if [ -x ./$app_dir/run_gui.sh ]; then
        cd ./$app_dir/
      ./run_gui.sh
fi
```

## 3.1 GUI Usage

### 3.1.1 Log in




- ✓ Login window is shown at start of DVR system,
- ✓ Other menus are disabled when login window is enabled
- ✓ Access level is divided 2 levels (ADMIN, USER).

- ✓ Maximum password character is 8
- ✓ Maximum user number is 10
- ✓ Default password is “blank” for both “ADMIN” and “USER”
- ✓ Password can be changed in *“Setup → System”* menu
- ✓ Access control for each mode (ADMIN, USER) can be selected in *“Setup → System”* menu

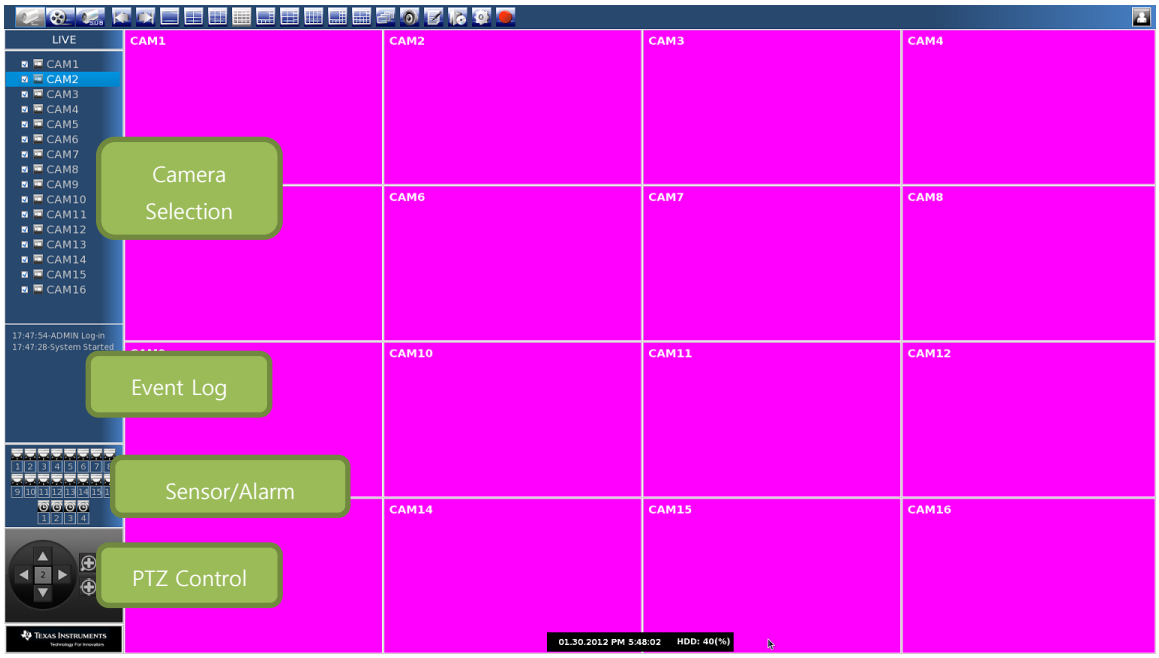
**Note:** Keyboard feature on the login box is not implemented in this release

### 3.1.2 Tool bar

The toolbar allows multiple functions based on the button press on the toolbar. Most of the buttons toggle the operation (enable or disable the feature).

- ✓  : Live mode
- ✓  : Playback mode
- ✓  : Sub-screen(secondary output) mode setting tool bar. This button pops up a reduced toolbar on the right bottom of the screen.
- ✓  : Previous / next page
- ✓  : screen mode setting
- ✓  : Sequence mode toggle (on/off). Sequence mode enables automatic display of next page after a programmable timeout interval. The timeout interval can be changed *“Setup → Display”* menu
- ✓  : Audio output enable/disable
- ✓  : Event log – this button show log pop-up dialog.
- ✓  : Back-up – This button results in popping up the backup configuration menu.
- ✓  : Setup – Clicking this button pops up the setup menu of the DVR
- ✓  : Emergency recording on/off – This button enables forced recording on all channels irrespective of the recording schedule. This is useful when the security administrator wants to override a recording in emergency situations.
- ✓  : log out button

### 3.1.3 Live



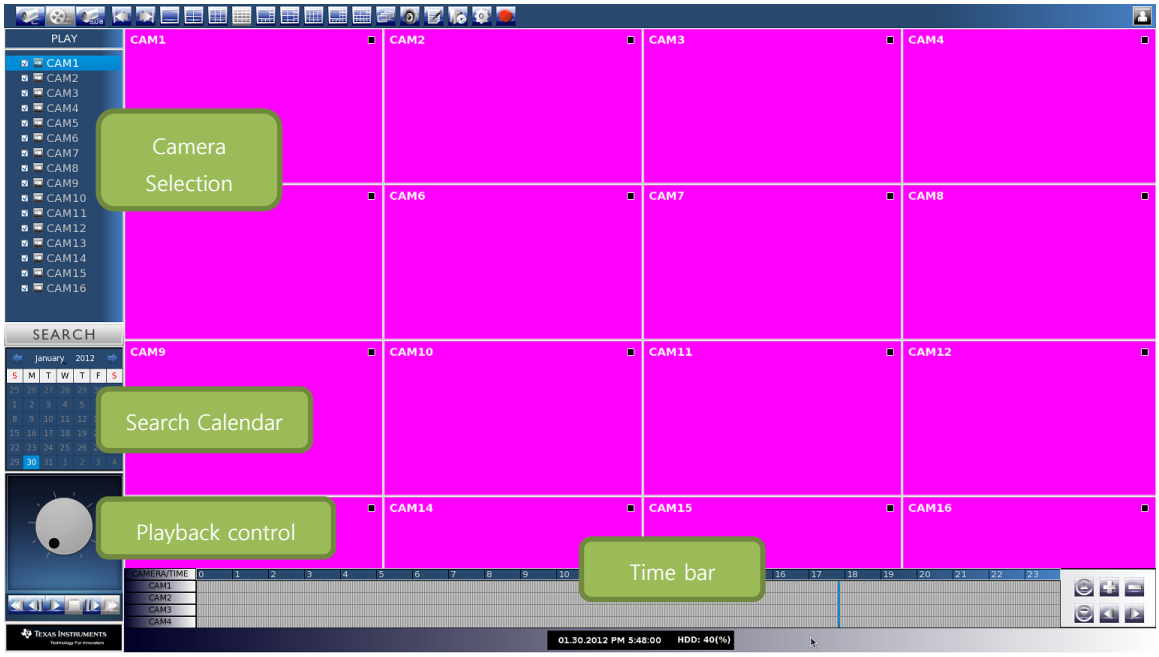

This is the default mode in which the DVR is started. The display shows a grid of multiple channels along with multiple control options. The control options are listed below:

- ✓ **Camera control window** – This window allows control the selection of input cameras on the display. A checkbox on the side of the camera enables or disables the preview of the camera on the Live Preview display.
  For example, If CAMERA1 is disabled on the camera control window, CAMERA2 moves to the position of CAMERA1 and all other cameras follow-on.
- ✓ **Event log window** – This window is used to show the logs of events or operations happening on the DVR
- ✓ **Sensor/Alarm window** – This window shows the status of sensors and alarms. The active sensors and alarms are shown with the enabled icon.
- ✓ **PTZ control window** – This window provides buttons to control camera pan, tilt and zoom.

The descriptions of PTZ buttons are as following:

- ✓  : move camera position (up/down/left/right)
- ✓  : zoom in/zoom out
- ✓  : focus in/focus out

### 3.1.4 Playback




This window is seen when the Playback mode is selected using the main toolbar. In the current release, the playback channels are displayed on the secondary output (sub-screen display). The secondary display shows a grid of multiple playback channels. The control options are listed below:

- ✓ **Camera control window** – This window allows control the selection of input cameras on the secondary display. A checkbox on the side of the camera enables or disables the display of the channel

  For example, If CAMERA1 is disabled on the camera control window, CAMERA2 moves to the position of CAMERA1 and all other cameras follow-on.

- ✓ **Calendar Search window** – This window shows a calendar. User can select the specific date to view the recording corresponding to that date. The days which have recorded data are shown in bold.
- ✓ **Time bar window** – This window shows all the recorded data for the selected date. The time bar allows the user to look at the recorded data for each channel at a particular time. It also shows the recording type (event, continuous, alarm…etc) using the color code. Color code for the different recording options is explained in "*Setup→Record"*
- ✓ **Playback control window** – This window allows the user to select the playback speed. The window has a virtual jog key that can be controlled using mouse interface.

On the right hand side of the time bar, there exists control buttons for the time bar. Descriptions of the buttons are listed below:

- ✓  : **channel up/down** – The time bar shows only 4 channels at a time. User can select the next set of 4 channels using these buttons.

- ✓  : **zoom in/out** – Time bar shows a range of 24 hours. User can use these zoom buttons to zoom in or out in the time interval. The time bar can zoom into a range of 6 hours to a range of 24 hours (default). For example, time bar shows 00:00 to 24:00 and if user zooms in, the time bar shows 00:00 to 06:00. This would allow user to select a better granularity for playback.
- ✓  : **move previous/next time band** – When time bar is selected to show 6 hours range, user can go left or right to select next time band. For example, if user has zoomed into 6 hours range and time bar shows 00:00 to 06:00, then user can move to 06:00 to 12:00 by clicking the right button.

### 3.1.5 Status bar


It shows the date/time depending on the live mode or playback mode selection.

- ✓ Live mode – In this mode, status bar shows current date/time
- ✓ Playback mode – In this mode, status bar shows the selected playback date/time

It also shows hard disk usage information.

### 3.1.6 Status Icon

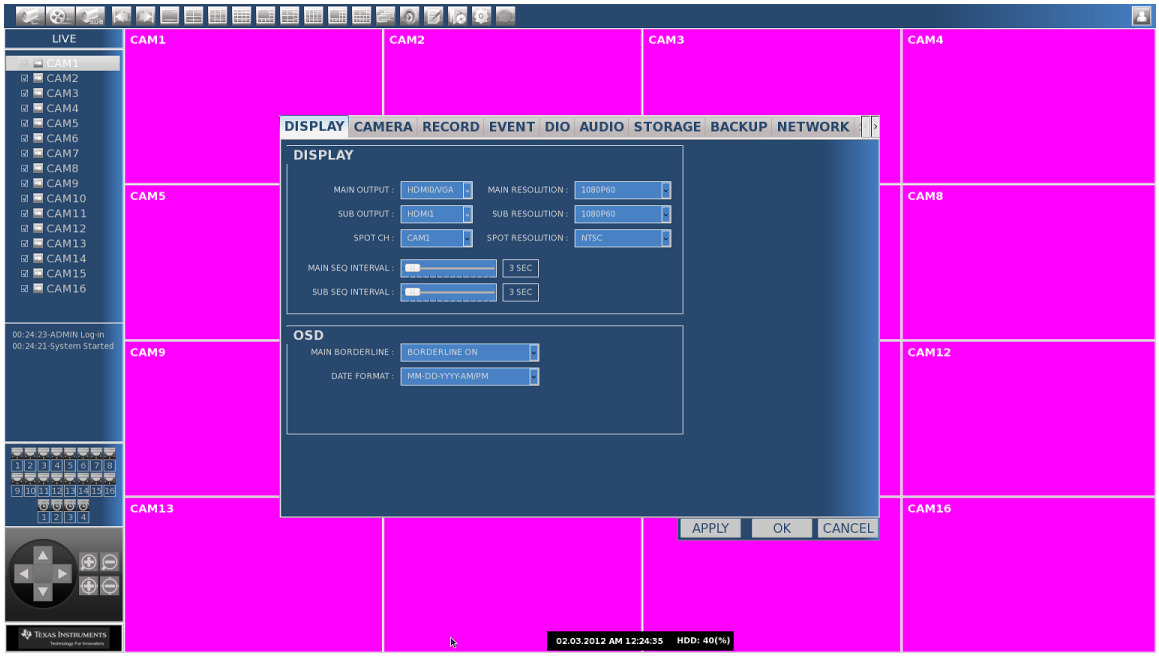

It shows the status of system with following symbols on OSD

- ✓  : shows the status of video loss detection. Whenever a camera input source is removed, the display goes blank for that camera and this icon shows up on that specific blank display.
- ✓  : shows motion detection. Whenever motion detection is enabled and there is motion detected in the selected region, this icon shows up on the display for that camera.
- ✓  : shows the status of audio recording. Whenever audio recording is enabled for that camera source, this icon shows up on the display for that camera.
- ✓  : shows the status of video recording. This icon shows up when recording for a

particular camera is on.

- ✓  : indicates playback status in playback mode

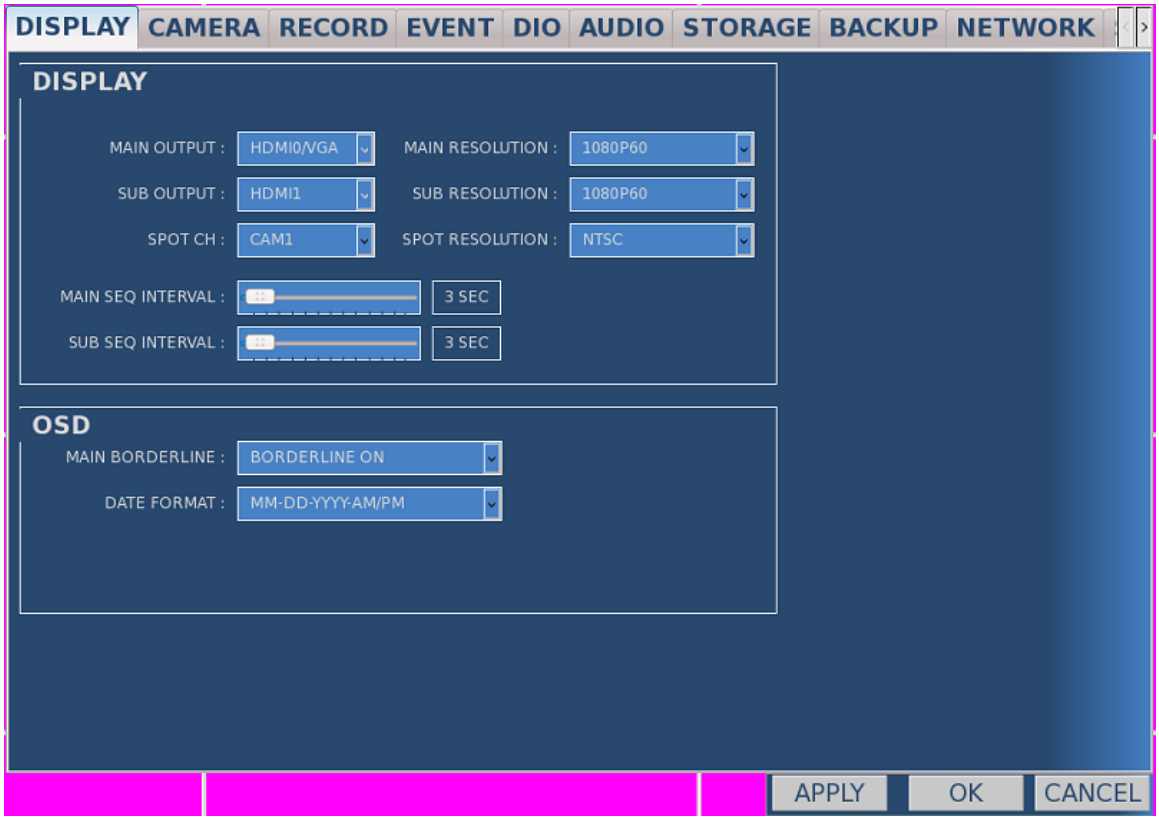
### 3.1.7 Setup menu




Setup Menu has multiple tabs for the different sub menus. The sub-menus and their key features are listed below:

- ✓ **Display setup** – Controls for output selection, resolution configuration, OSD controls
- ✓ **Camera setup** – Provides control for the individual input sources. Controls enable/disable, camera naming, motion detection, image filter, color, PTZ setting
- ✓ **Record setup** – video/audio recording setup, event record duration, record schedule
- ✓ **Event setup** – event to record, event to alarm setting
- ✓ **DIO setup** – sensor, alarm setting
- ✓ **Audio setup** – audio input/output
- ✓ **Storage setup** – HDD format/recycle, disk information
- ✓ **Backup** – provide backup functions
- ✓ **Network setup** – network type, IP, subnet mask, gateway for eth0 and eth1
- ✓ **System setup** – system version, date/time, user, system initialize and reboot

### 3.1.8 Display setup



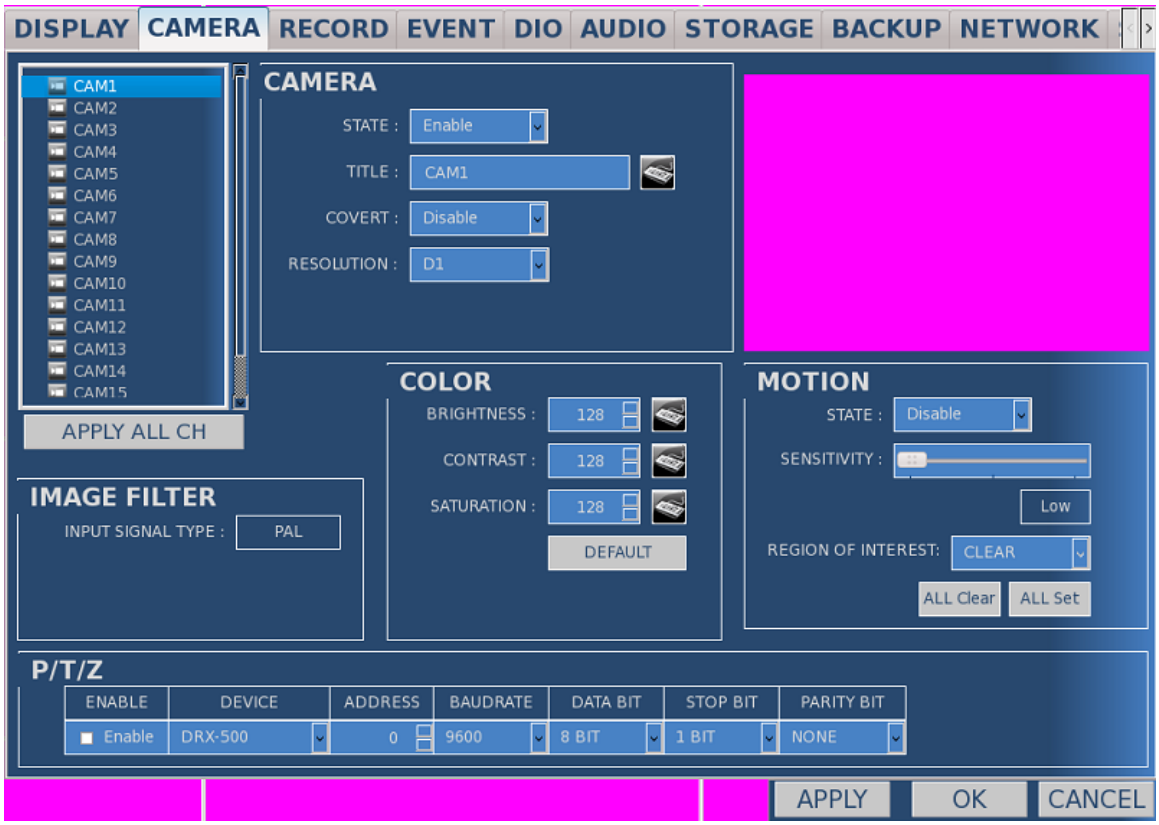

#### 3.1.8.1 Display

- ✓ **Main output** – HDMI0/VGA /HDMI1 – show current Main-output port.
- ✓ **Sub output** – HDMI0/VGA /HDMI1 – show current Sub-output port.
- ✓ **Spot ch** – Only one channel can be displayed on spot output. This is displayed selected channel on the CVBS composite output.
- ✓ **Main sequence interval** – selectable from 1 second to 60 seconds. Once sequencing is enabled using the main toolbar, the automatic page up/down happens after this interval of time.
- ✓ **Sub sequence interval** – selectable from 1 second to 60 seconds. Once sequencing is enabled using the main toolbar, the automatic page up/down happens after this interval of time.
- ✓ **Main/Sub Resolution** – XGA/SXGA/720P/1080P – select output resolution.
- ✓ **Spot resolution** – NTSC/PAL

#### 3.1.8.2 OSD

- ✓ **Main border line** – border line on/off for main output – These border lines are drawn on the display grid.
- ✓ **Date format** – MM-DD-YYYY-AM/PM / DD-MM-YYYY-AM/PM / YYYY-DD-MM-AM/PM / YYYY-MM-DD-AM/PM / MM-DD-YYYY / DD-MM-YYYY / YYYY-DD-MM / YYYY-MM-DD – User can select the date format that is to be displayed on status bar.

### 3.1.9 Camera setup




User can select the camera on the left tab and change the properties of the camera with the following menus.

APPLY ALL CH button – apply current camera setting except title to all camera.

#### 3.1.9.1 Camera

- ✓ **Enable** – camera enable/disable setting, it can also be done using the “Camera Tree Menu” on the left. When a camera is enabled, it is displayed on the live preview.
- ✓ **Title** – camera title setting. User can set the camera title that is displayed on the live preview.
- ✓ **Covert** – When this option is enabled, only encoding & recording happens for the camera without it being displayed on the live preview. This is a feature needed for secret recording.
- ✓ **Resolution** – D1/CIF/Half D1 – User can select the resolution in which they want to encode and store the stream on the local storage.

#### 3.1.9.2 Motion

- ✓ **Detect** – motion detection enable/disable setting – User can enable motion detection for a particular camera.
- ✓ **Sensitivity** – select from level 1(Low) to level 3(High) – Sensitivity of 1 is lowest and 3 is highest. This means if user sets sensitivity as 3(High), there is higher likelihood of detecting motion but can result in higher percentage of false detection as well.

- ✓ **Area mode** – selectable between clear/set – The region for motion detection can be selected on the right side. If the area mode is selected as “clear”, the motion detection area would be cleared. If the area mode is selected as “set”, the motion detection area would be selected on mouse click. “All Set” or “All Clear” buttons do the respective operation on the complete area of the camera input.

#### 3.1.9.3 Image filter

- ✓ **Input signal type** – NTSC/PAL – This is an input signal status display. If the input channels are NTSC, then this field shows NTSC, else PAL.

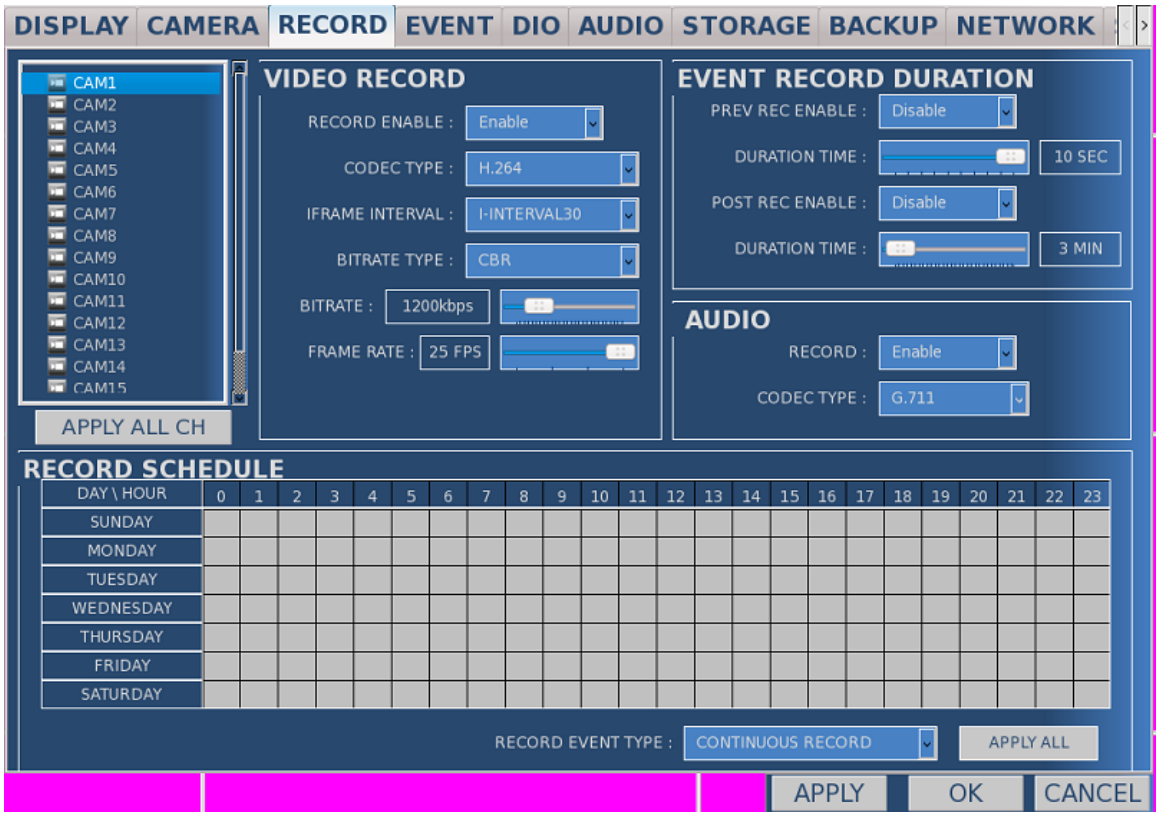
#### 3.1.9.4 Color

- ✓ **Brightness** – range 0 to 255 – Controls the brightness of a specific input channel.
- ✓ **Contrast** – range 0 to 255,
- ✓ **Saturation** – range 0 to 255,

#### 3.1.9.5 PTZ

This menu is to set the protocol for pan, tilt, and zoom. Once the protocol is selected for the camera, the actual pan, tilt and zoom can be controlled with the live preview PTZ control window.

### 3.1.10 Record setup




User can select the camera on the left tab and change the options of the recording with the

following menus.

APPLY ALL CH button – apply current record setting to all camera.

#### 3.1.10.1 Video record

- ✓ **Record enable** – recording enable/disable – If this menu option is enabled, the recording for the channel is started based on the recording type and schedule.
- ✓ **Codec type** – H264/MPEG4/MJPEG **(currently only H264 is supported)**
- ✓ **Iframe interval** – 1/5/10/15/30 – Possible selection of the IFrame Interval for each channel.
- ✓ **Bitrate type** – CBR/VBR – This controls the rate control algorithm for the compression. CBR would mean that the compression would maintain the bitrate and can possibly drop frames. VBR means that the bitrate can be variable but the quality remains constant.
- ✓ **Bitrate** – 500Kbps to 4000Kbps (increase value by the 100Kbps). User can control the bitrate of each stream.
- ✓ **Frame rate** – 4/8/15/30 for NTSC camera, 3/6/13/25 for PAL camera.

#### 3.1.10.2 Event record duration

- ✓ **Prev Rec Enable** - When enabled, system would record the video prior to the event.
  - ■ **Duration time** – User can set the duration of the recording prior to the event.
- ✓ **Post Rec Enable** - When enabled, system would record after the event is detected.
  - ■ **Duration time** – User can set the duration of the recording post the event.

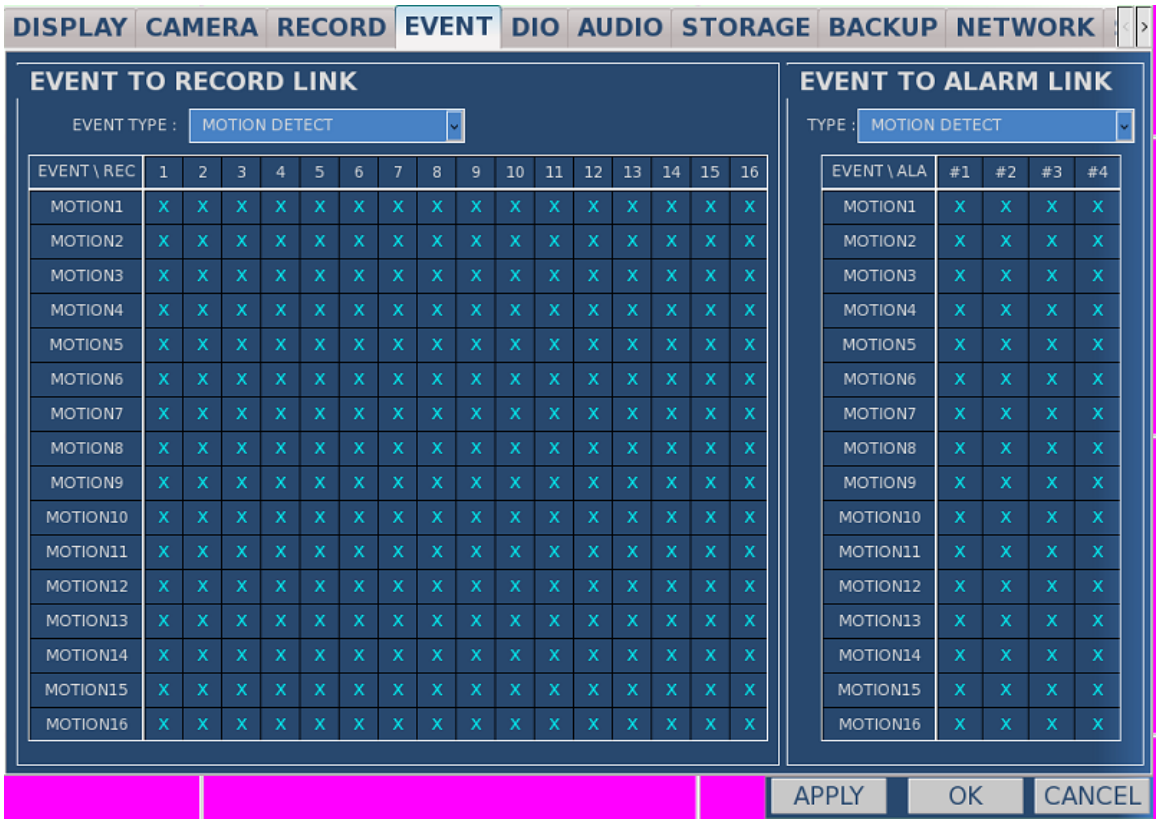
#### 3.1.10.3 Audio

- ✓ **Record enable** – audio recording enable/disable,
- ✓ **Codec type** – G711/AAC, **(currently only G711 is supported)**

#### 3.1.10.4 Record schedule

- ✓ **Record event type** – continuous record / record by motion / record by sensor / record by video loss / no record – User can plan a recording schedule using this menu. User can first select the event to be used for triggering the recording and then set up the schedule by clicking on the grid. The schedule can be selected for the whole week. User has an option of clicking “Apply All” button which results in the complete schedule set for the selected recording event. Option of “no record” is available in case user does not want any recording and would enable recording using emergency record mode (forcible recording).

### 3.1.11 Event setup



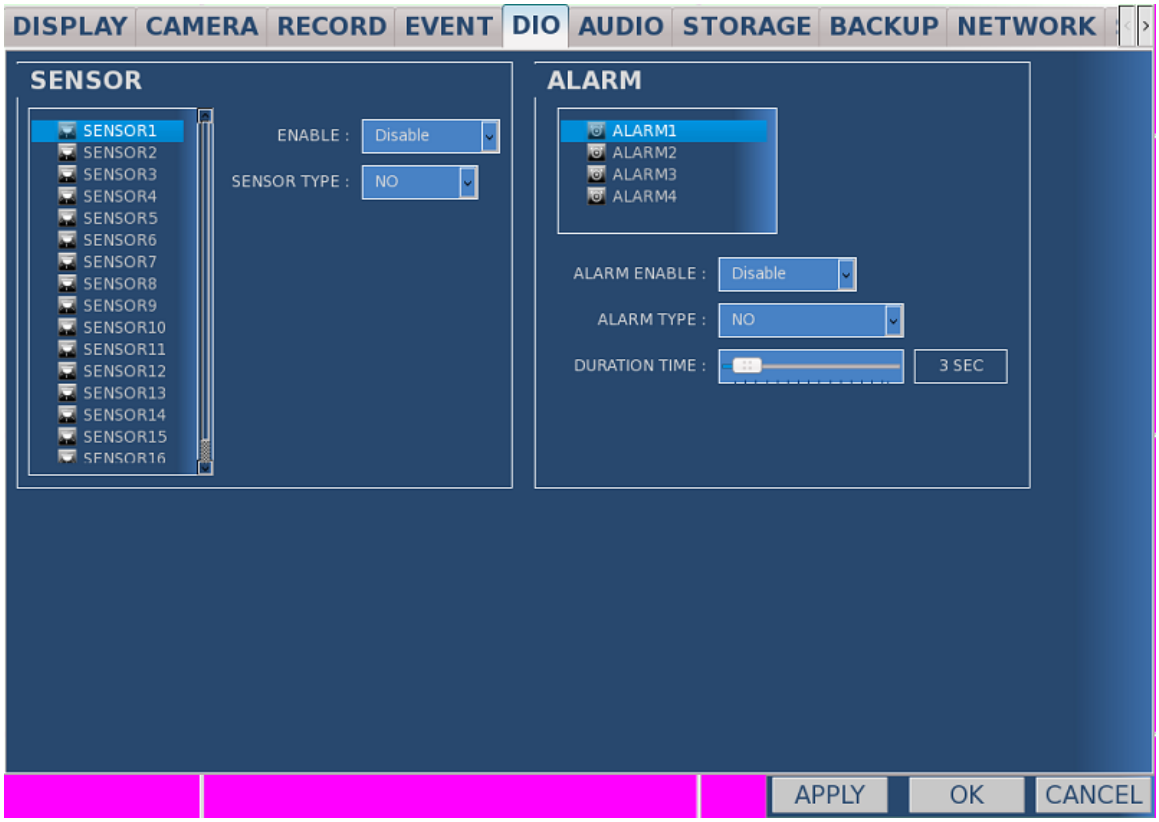

#### 3.1.11.1 Event to record link

- ✓ Event type – motion/sensor/video loss/, not implemented about motion and video loss in this release

#### 3.1.11.2 Event to alarm link

- ✓ Type – motion/sensor/video loss/, not implemented about motion and video loss in this release

### 3.1.12 DIO setup



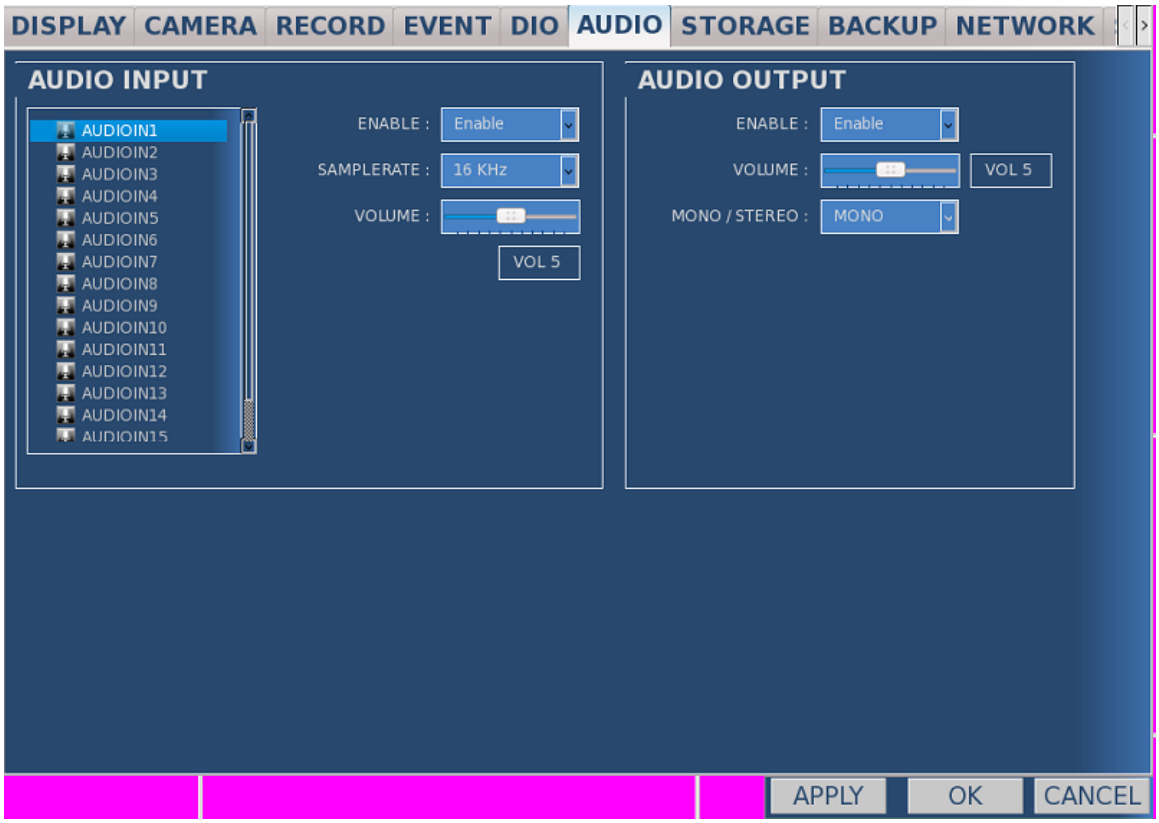

#### 3.1.12.3 Sensor

- ✓ Enable – enable/disable for each sensor
- ✓ Sensor type – NO/NC(Normal Open / Normal Close)

#### 3.1.12.4 Alarm

- ✓ Alarm enable – enable/disable for each alarm
- ✓ Alarm type – NO/NC,
- ✓ Duration time – can change the duration of alarm from 1 second to 30 seconds

### 3.1.13 Audio setup




User can select one of the 16 audio input channels on the left tab and do the configuration for that channel using following menus.

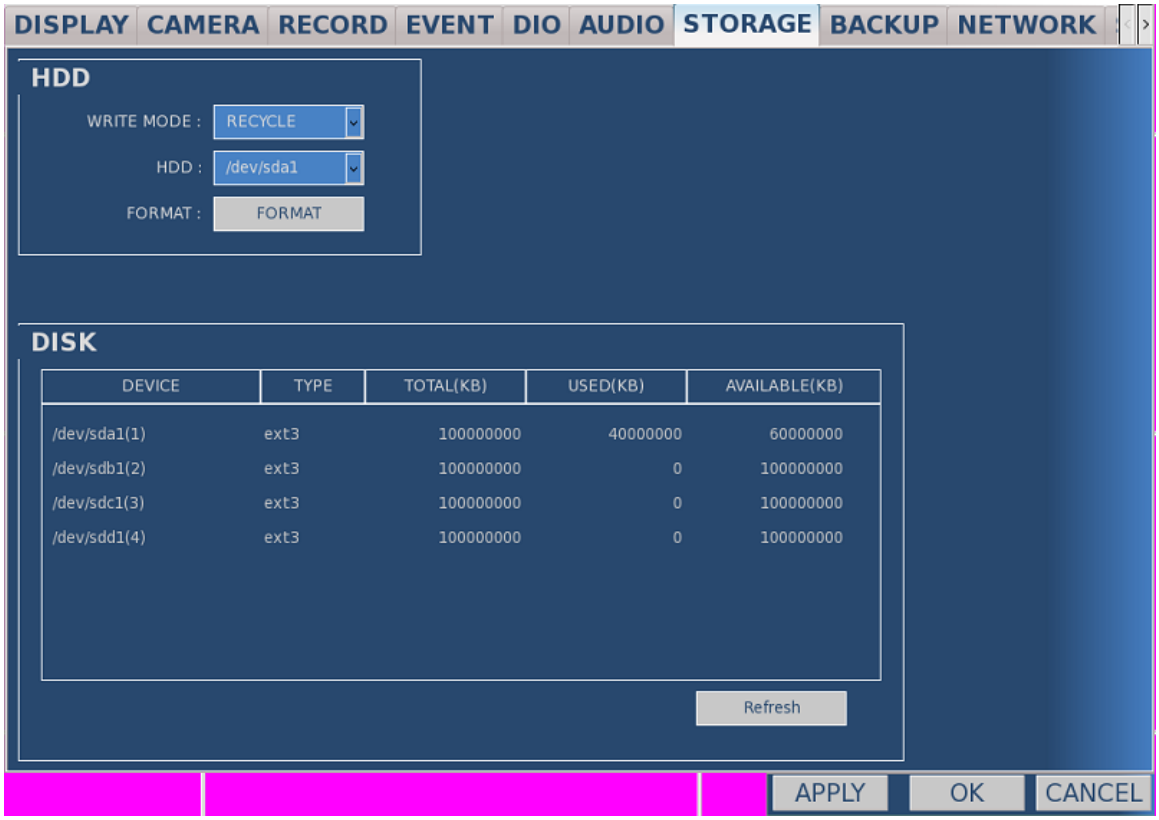
#### 3.1.13.1 Audio input

- ✓ **Enable** – enable/disable for each audio input
- ✓ **Sample rate** – 8KHz/16KHz,
- ✓ **Volume** – mute to 10 – volume of 1 is lowest and 10 is highest **(not implemented in this release)**

#### 3.1.13.2 Audio output

- ✓ **Enable** – enable/disable for audio output,
- ✓ **Volume** – mute to 10 – volume of 1 is lowest and 10 is highest **(not implemented in this release)**

### 3.1.14 Storage setup




#### 3.1.14.3 HDD

- ✓ **Write mode** – recycle/once. It is setting for HDD operation, when HDD is full.
  Once – After the HDD is full, no further recording allowed on the hard disk
  Recycle –After HDD is full, the oldest files start getting deleted automatically
- ✓ **HDD** – shows device path for the active hard disk and select device for format disk
- ✓ **Format** – format the selected HDD and created Basket File System.
  If have not created a partition on HDD, it will run Fdisk automatically and then format proceeds

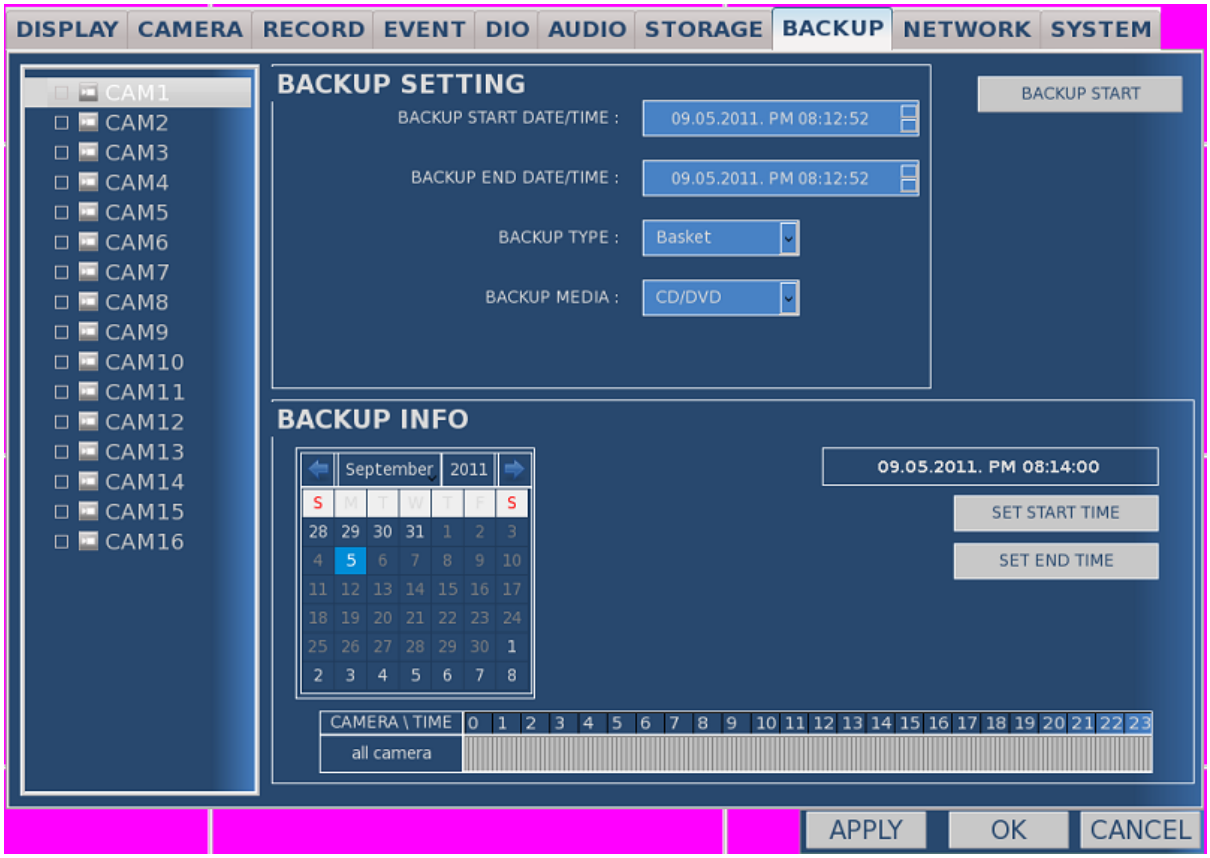
Note : Format command of the RDK include formatting HDD and creating a Basket File System for recording, so it is available only for recording of the video data in the RDK. It supports up to four HDD for recording. After the format of the DEVICE, number (1) to (4) is appended next to the disks. This means the HDDs are recordable. Note: (0) means that disk is not recordable.

#### 3.1.14.4 Disk

This table shows the entire mounted disk's information.

Note :**TOTAL**, **USED**, **AVAILABLE** sizes in the Disk information matches the disk information after running '**df**' command on Linux shell

## 3.1.15 Backup setup




### 3.1.15.1 Camera selection

Camera selection is enabled when backup type is AVI. User can select specific camera for backup

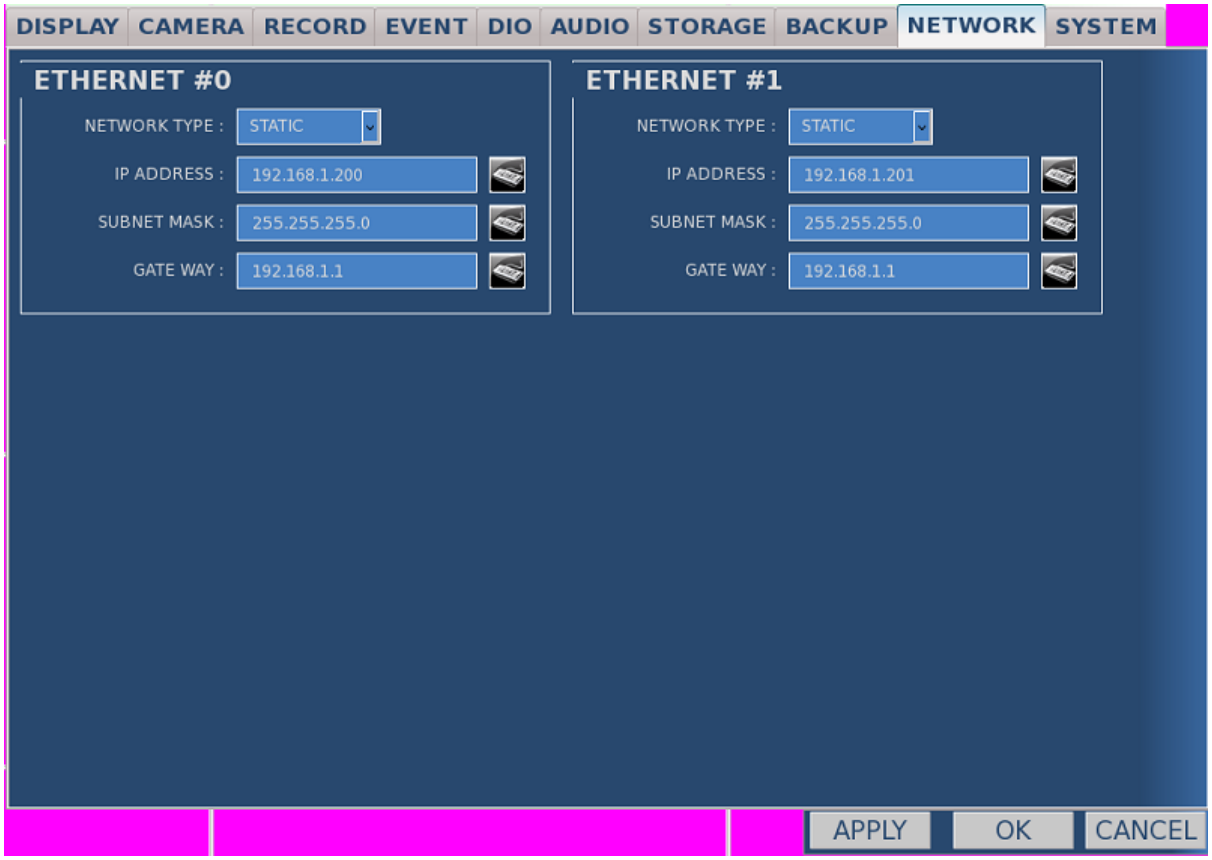
### 3.1.15.2 Backup setting

- ✓ Backup start date/time – can select start date/time
- ✓ Backup end date/time – can select end date/time
- ✓ Backup type – can select basket/AVI
- ✓ Backup media – can select CD/DVD or USB
- ✓ Select device – when backup media is USB, user should select device path

### 3.1.15.3 Backup info

This show recording status of selected camera and provide start/end time setting

### 3.1.16 Network setup




- ✓ **Network type** – STATIC/DHCP
- ✓ IP address
- ✓ Subnet mask
- ✓ Gateway

User can set each of the values for IP address, subnet mask and gateway address for the two Ethernet interfaces on the DVR RDK. If the DHCP mode is selected, the other addresses are not valid.

### 3.1.17 System Setup


#### 3.1.17.1 System information

This menu displays the status of the system for the following items:

- ✓ Firmware version
- ✓ Hardware version
- ✓ MAC address#1
- ✓ MAC address#2
- ✓ Build date/time

#### 3.1.17.2 Date/time

- ✓ Current – shows current system time
- ✓ Config – update system time – User can input a new date/time for the system and that gets reflected in current time.

#### 3.1.17.3 System Config

- ✓ Config initialize – initializes system to default factory configurations. All the items in setup menus go to default values. This option can be used to recover from some unknown condition.
- ✓ System reboot – reboot DVR system – User can click this button to auto reboot the DVR and start up fresh.

#### 3.1.17.4 User

Select the user name from the left tab and configure the user properties using the following options:

- ✓ User enable – enable/disable access for the selected user
- ✓ Access content – give access right for live/playback/main menu/setup
- ✓ Change passwd – User has to enter the new password
- ✓ Confirm passwd – User has to enter the new password again for confirmation

## 3.2 User API Interface

- ✓ The API interface header files are present at <DVR_RDK_INSTALL_DIR>/dvr_rdk/dvrapp /app/inc
    - app_manager.h – API include file

### 3.2.1 LIB816x_backupToAVI

| | | | |
|---|---|---|---|
| **Description** | Start AVI backup function. | | |
| **Prototype** | *int LIB816x_backupToAVI(int media, char *path, int ch_bitmask, struct tm start_t, struct tm end_t)* | | |
| **Arguments** | media | [IN] | target media. 0 : CDROM, 1: USB |
| | path | [IN] | Media path |
| | ch_bitmask | [IN] | Backup channel bitmask |
| | start_t | [IN] | Start time |
| | end_t | [IN] | End time |
| **Return** | int | 0(LIB_BA_NO_ERR) if succeed, -3(LIB_BA_BASKET_FAILED) if failed | |

### 3.2.2 LIB816x_backupToBASKET

| | | | |
|---|---|---|---|
| **Description** | Start BASKET file backup function. | | |
| **Prototype** | *int LIB816x_backupToBASKET(int media, char *path, int ch_bitmask, struct tm start_t, struct tm end_t)* | | |
| **Arguments** | media | [IN] | target media. 0 : CDROM, 1: USB |
| | path | [IN] | Media path |
| | ch_bitmask | [IN] | Backup channel bitmask |
| | start_t | [IN] | Start time |
| | end_t | [IN] | End time |
| **Return** | int | 0(OSA_SOK) if succeed, -1(OSA_EFAIL) if failed | |

### 3.2.3 LIB816x_BasketCreate

**Description** Create basket files with given target path.

**Prototype** *int LIB816x_BasketCreate(char *mntpath)*

**Arguments** mntpath [IN] target media mounted path

**Return** int Count of created basket files. On error return 0.

### 3.2.4 LIB816x_BasketInfo

**Description** Get basket count and size on basket system.

**Prototype** *int LIB816x_BasketInfo(long *bkt_count, long *bkt_size)*

**Arguments**

| | | |
|---|---|---|
| bkt_count | [OUT] | Total count of basket files. |
| bkt_size | [OUT] | A basket size as Mega Bytes. |

**Return** int 1 if succeed, -1 if failed

### 3.2.5 LIB816x_CDROM_EJECT

**Description** Eject CD-ROM.

**Prototype** *int LIB816x_CDROM_EJECT(char *device)*

**Arguments** device [IN] Device name

**Return** int 1

### 3.2.6 LIB816x_CDROM_ERASE

**Description** Erase CD-ROM.

**Prototype** *int LIB816x_CDROM_ERASE(char *device)*

**Arguments** device [IN] Device name

**Return** int 1

### 3.2.7 LIB816x_CDROM_MAKE_ISO

**Description** Make ISO file

**Prototype** *int LIB816x_CDROM_MAKE_ISO(char *iso_name, char **filelist, int filecnt)*

| | | | |
|---|---|---|---|
| | iso_name | [IN] | Channel number |
| **Arguments** | filelist | [IN] | File list |
| | filecnt | [IN] | File count |
| **Return** | int | 1 | |

### 3.2.8 LIB816x_CDROM_MEDIA

**Description** Check ready media or no media.

**Prototype** *int LIB816x_CDROM_MEDIA(char *device)*

| | | | |
|---|---|---|---|
| **Arguments** | device | [IN] | Device name |
| **Return** | int | 1 if no media or not ready, 0 if CDR, 2 if DVD | |

### 3.2.9 LIB816x_CDROM_WRITE_ISO

**Description** Write ISO file to CD

**Prototype** *int LIB816x_CDROM_WRITE_ISO(char *device, char *iso_file)*

| | | | |
|---|---|---|---|
| **Arguments** | device | [IN] | Device name |
| | iso_file | [IN] | iso file name |
| **Return** | int | 1 if succeed, 0 if failed | |

### 3.2.10 LIB816x_changeCamName

**Description** Change camera name

**Prototype** *int LIB816x_changeCamName(int chId, char* camName)*

| | | | |
|---|---|---|---|
| **Arguments** | chId | [IN] | Channel number |
| | camName | [IN] | Camera name |
| **Return** | int | 1 | |

### 3.2.11 LIB816x_disk_info

**Description** Get disk information

**Prototype** *int LIB816x_disk_info(void *disk_info)*

| | | | |
|---|---|---|---|
| **Arguments** | disk_info | [OUT] | Disk information. Refer to struct dvr_disk_info_t |
| **Return** | int | 0 if succeed, -1 if failed | |

### 3.2.12 LIB816x_disk_size

| | | | |
|---|---|---|---|
| **Description** | Get disk size | | |
| **Prototype** | *int LIB816x_disk_size(char *mntpath, unsigned long *total, unsigned long *used)* | | |
| **Arguments** | mntpath | [IN] | Mounted path |
| | total | [OUT] | Total size. |
| | used | [OUT] | Used size. |
| **Return** | int | 0 if succeed, -1 if failed. | |

### 3.2.13 LIB816x_endCamProperty

| | |
|---|---|
| **Description** | Stop the PIP display |
| **Prototype** | *void LIB816x_endCamProperty()* |
| **Arguments** | none |
| **Return** | void |

### 3.2.14 LIB816x_fastBackward_x

| | | | |
|---|---|---|---|
| **Description** | Set fast backward playback | | |
| **Prototype** | *void LIB816x_fastBackward_x (int ch_bitmask)* | | |
| **Arguments** | ch_bitmask | [IN] | Channel bitmask |
| **Return** | int | 0 if no error, -7(LIB_PB_NOT_ALLOWED) if not open basket. | |

### 3.2.15 LIB816x_get_encoder_property

| | |
|---|---|
| **Description** | According given *type*, get encoder property. Framerates , bitrate, i-frame interval or resolution. |

**Prototype** *int LIB816x_get_encoder_property(int type, int channel_bitmask, int *pValue)*

**Arguments**

| | | |
|---|---|---|
| *type* | [IN] | kind of get value. LIB_ENC_GET_FRAMERATE, LIB_ENC_GET_BITRATE, LIB_ENC_GET_I_FRAME_INTERVAL, LIB_ENC_GET_RESOLUTION, LIB_ENC_GET_REQ_KEY_STATUS |
| *channel_bitmask* | [IN] | Channel bitmask of get value |
| *pValue* | [OUT] | Framerate, Bitrate, I-frame Interval, Resolution |

**Return** int 0 if no error, -7(LIB_PB_NOT_ALLOWED) if not open basket.

### 3.2.16 LIB816x_getAlarmStatus

**Description** Get alarm status. MAX_ALARM.

**Prototype** *void LIB816x_getAlarmStatus ()*

**Arguments** None

**Return** unsigned int Bitmask of alarm status flag

### 3.2.17 LIB816x_getColorAdjust

**Description** Get Color Adjustment. Like a Brightness, Contrast and Saturation

**Prototype** *int LIB816x_getColorAdjust(int nChannelIndex, COLORADJUST* padjust);*

**Arguments**

| | | |
|---|---|---|
| nChannelIndex | [IN] | Channel number |
| padjust | [OUT] | Adjustments(Contrast, Saturation, Brightness) |

**Return** int 0 if succeed, -1 if failed.

### 3.2.18 LIB816x_getCurPlaybackTime

**Description** Get current playback time

**Prototype** *int LIB816x_getCurPlaybackTime(struct tm *tp)*

**Arguments** tp [OUT] Current play time.

**Return** int 1

### 3.2.19 LIB816x_getDvrMessage

**Description** Read message from pipe

**Prototype** *int LIB816x_getDvrMessage(DVR_MSG_T *pMsg)*

**Arguments** pMsg [OUT] Current play time.

**Return** int OSA_SOK if succeed, OSA_EFAIL if failed

### 3.2.20 LIB816x_getIntPtzCount

**Description** Get the embedded ptz control list.

**Prototype** *int LIB816x_getIntPtzCount()*

**Arguments** None

**Return** int INTERNAL_PTZ_COUNT(2), fixed

### 3.2.21 LIB816x_getIntPtzInfo

**Description** Get PTZ name by given index number

**Prototype** *int LIB816x_getIntPtzInfo(int ptzIdx, char* pPtzName)*

**Arguments**

| | | |
|---|---|---|
| ptzIdx | [IN] | PTZ index to get name |
| pPtzName | [OUT] | destination of String copy |

**Return** int 1 if succeed, 0 if failed.

### 3.2.22 LIB816x_GetLastRecTime

**Description** Get latest record seconds from record data(RDB)

**Prototype** *int LIB816x_GetLastRecTime ()*

**Arguments** None

**Return** long Latest record seconds if succeed, -1 if failed.

### 3.2.23 LIB816x_getMotionDetectStatus

**Description** Get motion detect status flags for all channels

**Prototype** *int LIB816x_getMotionDetectStatus ()*

**Arguments** None

**Return** unsigned int Bitmask of motion detect flag

### 3.2.24 LIB816x_GetRecDays

**Description** Get record days by given time(year and month)

**Prototype** *long LIB816x_GetRecDays(struct tm t, int* pRecDayTBL)*

**Arguments**

| | | |
|---|---|---|
| t | [IN] | Particular year and month. |
| pRecDayTBL | [OUT] | Record days data array.(31 days) |

**Return** long 1 if found, 0 if failed.

### 3.2.25 LIB816x_GetRecHour

**Description** Get record hours by given time(year, month and day)

**Prototype** *long LIB816x_GetRecHour(int ch, struct tm t, int* pRecHourTBL)*

**Arguments**

| | | |
|---|---|---|
| ch | [IN] | Particular channel |
| t | [IN] | Particular year, month and day |
| pRecHourTBL | [OUT] | Record hours data array(24hourX60min=1440min) |

**Return** long 0 fixed.

### 3.2.26 LIB816x_getSensorStatus

**Description** Get sensor status flags for all sensor(MAX_SENSOR)

**Prototype** *int LIB816x_getSensorStatus ()*

**Arguments** None

| | | | |
|---|---|---|---|
| **Return** | unsigned int | Bitmask of sensor status flag | |

### 3.2.27 LIB816x_getSourceStatus

| | | | |
|---|---|---|---|
| **Description** | Get the input camera resolution. | | |
| **Prototype** | *int LIB816x_getSourceStatus(int *nChannelCount, SOURCE_CH_STATUS_S* pChStatus)* | | |
| **Arguments** | nChannelCount | [OUT] | Captured channel count |
| | pChStatus | [OUT] | Source channel status(SOURCE_CH_STATUS_S) |
| **Return** | long | 0 fixed. | |

### 3.2.28 LIB816x_getVideoLossDetectStatus

| | | | |
|---|---|---|---|
| **Description** | Get video loss status flags for all video source status number channels | | |
| **Prototype** | *unsigned int LIB816x_getVideoLossDetectStatus ()* | | |
| **Arguments** | None | | |
| **Return** | unsigned int | Bitmask of Video detected status flag | |

### 3.2.29 LIB816x_HddFormat

| | | | |
|---|---|---|---|
| **Description** | HDD format | | |
| **Prototype** | *int LIB816x_HddFormat (char *mntpath, char *devpath, int hddstatus)* | | |
| **Arguments** | mntpath | [IN] | mount directory |
| | devpath | [IN] | device name ( ex, /dev/sda1) |
| | hddstatus | [IN] | Format step. DISK_IDLE, DONE_FDISK, DONE_EXT3, DONE_FORMAT |
| **Return** | int | 1 if succeed, 0 if failed. | |

### 3.2.30 LIB816x_initDisplayInfo

| | |
|---|---|
| **Description** | N/A. Initialize diplay information. |

**Prototype** *void LIB816x_initDisplayInfo(ST_DISPLAY_PROPERTY *pDspProperty)*

**Arguments** *pDspProperty* [IN]

**Return** void

### 3.2.31 LIB816x_initPlayback_x

**Description** Create playback task main thread

**Prototype** *int LIB816x_initPlayback_x (void)*

**Arguments** *None*

**Return** int 0 fixed.

### 3.2.32 LIB816x_initSettingParam

**Description** Copy given pParam to global gInitSettings variable.

**Prototype** *void LIB816x_initSettingParam(SETTINGPRM* pParam);*

**Arguments** pParam [IN] Source setting data(SETTINGPRM)

**Return** void

### 3.2.33 LIB816x_jumpPlayback_x

**Description** Jump play point by given time and channel

**Prototype** *int LIB816x_jumpPlayback_x(int ch_bitmask, struct tm *ptm)*

**Arguments**

ch_bitmask [IN] Channel bitmask

ptm [IN] Play jump time

**Return** int 0 if succeed, -4 if failed

### 3.2.34 LIB816x_net_info

**Description** Set or Get network info

**Prototype** *int LIB816x_net_info(int set, int devno, void *net_info)*

| | | | |
|---|---|---|---|
| | set | [IN] | Set or Get Operation flag |
| **Arguments** | devno | [IN] | Network device number |
| | net_info | [IN/OUT] | dvr_net_info_t |
| **Return** | int | 0 if succeed, others if failed | |

### 3.2.35 LIB816x_operateAlarm

**Description** Send alarm on/off

**Prototype** *void LIB816x_operateAlarm(int iAlarmId,int iAlarmOnOff)*

| | | | |
|---|---|---|---|
| **Arguments** | *iAlarmId* | [IN] | Alarm index number |
| | *iAlarmOnOff* | [IN] | Flag of on or off |
| **Return** | void | | |

### 3.2.36 LIB816x_pausePlayback_x

**Description** Pause playback

**Prototype** *int LIB816x_pausePlayback_x(void)*

**Arguments** none

**Return** int 0 fixed.

### 3.2.37 LIB816x_ptzCtrl

**Description** Control internal PTZ

**Prototype** *int LIB816x_ptzCtrl(int ptzIdx, int ptzTargetAddr,int ctrl)*

| | | | |
|---|---|---|---|
| | ptzIdx | [IN] | index of internal PTZ LIST |
| **Arguments** | *ptzTargetAddr* | [IN] | Target address(id) |
| | ctrl | [IN] | Control command, RIGHT, LEFT, UP, and etc |
| **Return** | int | 0 fixed. | |

### 3.2.38 LIB816x_ptzSendBypass

**Description** Send PTZ Control by pass

**Prototype** *int LIB816x_ptzSendBypass(char* ptzBuf, int bufSize)*

| | | | |
|---|---|---|---|
| **Arguments** | ptzBuf | [IN] | buffer |
| | *bufSize* | [IN] | Buffer size |
| **Return** | int | | Send size |

### 3.2.39 LIB816x_rec_disk_size

**Description** Get record disk size

**Prototype** *int LIB816x_rec_disk_size(unsigned long *total, unsigned long *used)*

| | | | |
|---|---|---|---|
| **Arguments** | total | [OUT] | Total disk size for all mounted disk |
| | *used* | [OUT] | Used size for all mounted disk |
| **Return** | int | | 0 fixed. |

### 3.2.40 LIB816x_restartPlayback_x

**Description** Restart playback with given channel bitmask

**Prototype** *int LIB816x_restartPlayback_x(int ch_bitmask)*

| | | | |
|---|---|---|---|
| **Arguments** | ch_bitmask | [IN] | Channel bitmask to play |
| **Return** | int | | 0 fixed. |

### 3.2.41 LIB816x_selCamPropCh

**Description** Select PIP channel id

**Prototype** *void LIB816x_selCamPropCh(int nCh)*

| | | | |
|---|---|---|---|
| **Arguments** | nCh | [IN] | channel |
| **Return** | void | | |

### 3.2.42 LIB816x_set_encoder_property

**Description** Set encoder properties.

**Prototype** *int LIB816x_set_encoder_property(int type, int channel_bitmask, int *pValue)*

| | | | |
|---|---|---|---|
| **Arguments** | Type | [IN] | Kind of property, FRAMERATE, BITRATE and etc |
| | channel_bitmask | [IN] | Channel bitmask to set property |
| | pValue | [IN] | Property(FRAMERATE, BITRATE and etc) |

**Return** int 0 fixed

### 3.2.43 LIB816x_setAlarm

**Description** Set alarm configuration

**Prototype** *void LIB816x_setAlarm(int iAlarmId,int iAlarmEnable,int iAlarmType,int iAlarmDelay)*

| | | | |
|---|---|---|---|
| **Arguments** | iAlarmId | [IN] | Alarm index(id) |
| | iAlarmEnable | [IN] | Enable or Disable |
| | iAlarmType | [IN] | Type of alarm |
| | iAlarmDelay | [IN] | delay |

**Return** void

### 3.2.44 LIB816x_setAudioCodecType

**Description** Set audio codec type. N/A

**Prototype** *int LIB816x_setAudioCodecType(int Ch,int iCodecType)*

| | | | |
|---|---|---|---|
| **Arguments** | Ch | [IN] | Channel number |
| | iCodecType | [IN] | Codec type ( AUDIO_CODEC_G711) |

**Return** int 0 fixed

### 3.2.45 LIB816x_setAudioInput

**Description** Set audio input enable/disable. N/A

**Prototype** *int LIB816x_setAudioInput(int Ch,int bEnable)*

| | | | |
|---|---|---|---|
| **Arguments** | Ch | [IN] | Channel number |
| | bEnable | [IN] | Enable or disable flag |

**Return** int 0 fixed

### 3.2.46 LIB816x_setAudioInputParams

**Description** Set audio input parameters. N/A

**Prototype** *int LIB816x_setAudioInputParams(int iSampleRate, int iVolume)*

| | | | |
|---|---|---|---|
| **Arguments** | iSampleRate | [IN] | Sampling rate |
| | iVolume | [IN] | Volume |

**Return** int 0 fixed

### 3.2.47 LIB816x_setAudioOutput

**Description** Set audio output configuration.

**Prototype** *int LIB816x_setAudioOutput(int bEnable, int iVolume, int bStereo, int iInputCh)*

| | | | |
|---|---|---|---|
| | bEnable | [IN] | On/Off |
| **Arguments** | iVolume | [IN] | Volume |
| | bStereo | [IN] | Whether stero or mono |

**Return** int 0 fixed

### 3.2.48 LIB816x_setBitrateType

**Description** Set bitrate type CBR or VBR with given channel

**Prototype** *int LIB816x_setBitrateType(int chId, int bitrateType)*

| | | | |
|---|---|---|---|
| **Arguments** | chId | [IN] | Channel |
| | bitrateType | [IN] | CBR or VBR |

**Return** int 0 fixed

### 3.2.49 LIB816x_setCameraEnable

**Description** Set camera display enable or disable

**Prototype** *int LIB816x_setCameraEnable(int iOutput, int nChannelIndex, int bEnable)*

| **Arguments** | | | |
|---|---|---|---|
| | iOutput | [IN] | Live or playback |
| | nChannelIndex | [IN] | Channel index |
| | bEnable | [IN] | Enable or Disable |

**Return** int 0 if succeed, -1 failed

### 3.2.50 LIB816x_setCameraLayout

**Description** Set display screen mode

**Prototype** *int LIB816x_setCameraLayout(int iOutput, int nStartChannelIndex, int iLayoutMode)*

| **Arguments** | | | |
|---|---|---|---|
| | iOutput | [IN] | Live or playback |
| | nStartChannelIndex | [IN] | Start channel index |
| | iLayoutMode | [IN] | Layout mode |

**Return** int 0 if succeed, others failed

### 3.2.51 LIB816x_setColorAdjust

**Description** Set Color Adjustment.

**Prototype** *int LIB816x_setColorAdjust(int nChannelIndex, COLORADJUST* padjust);*

| **Arguments** | | | |
|---|---|---|---|
| | nChannelIndex | [IN] | Channel number |
| | padjust | [IN] | Adjustments(Contrast, Saturation, Brightness) |

**Return** int 0 if succeed, -1 if failed.

### 3.2.52 LIB816x_setCovert

**Description** Set Covert mode by given channel.

**Prototype** *int LIB816x_setCovert(int nChannelIndex, int bEnable)*

| | | | |
|---|---|---|---|
| **Arguments** | nChannelIndex | [IN] | Channel number |
| | bEnable | [IN] | Flag of enable or disable |

**Return** int 0 if succeed, others if failed.

### 3.2.53 LIB816x_setDIOCallback

**Description** Set DIO callback function

**Prototype** *void LIB816x_setDIOCallback(void* fncb)*

| | | | |
|---|---|---|---|
| **Arguments** | fncb | [IN] | Callback function pointer |

**Return** void

### 3.2.54 LIB816x_setDisplayLayout

**Description** Set display layout

**Prototype** *int LIB816x_setDisplayLayout(int mode, int nChannelIndex)*

| | | | |
|---|---|---|---|
| **Arguments** | mode | [IN] | Layout mode |
| | nChannelIndex | [IN] | N/A |

**Return** int 1 fixed

### 3.2.55 LIB816x_setDisplayMainSub

**Description** Switch display between Main and Sub monitor.

**Prototype** *int LIB816x_setDisplayMainSub(int mainDevId, int subDevId)*

| | | | |
|---|---|---|---|
| **Arguments** | mainDevId | [IN] | Main dsp id(Live) |
| | subDevId | [IN] | Sub dsp id |

**Return** int 0 fixed

### 3.2.56 LIB816x_setDisplayRes

**Description** Set output display resolution.

**Prototype** *int LIB816x_setDisplayRes(int devId, int resType)*

**Arguments**

| | | |
|---|---|---|
| devId | [IN] | displayId(HDMI, COMP, DVO2, SD) |
| resType | [IN] | Resolution type(1080i, 1080p, and etc) |

**Return** int 0 succeed, -1 if failed.

### 3.2.57 LIB816x_setMotion

**Description** Set Motion data.

**Prototype** *int LIB816x_setMotion( int Ch, int bEnable, int sensitivity, unsigned char* motionTable)*

**Arguments**

| | | |
|---|---|---|
| Ch | [IN] | Channel number |
| bEnable | [IN] | Enable flag |
| sensitivity | [IN] | Sensitivity level |
| motionTable | [IN] | Motion flags table (MAX_MOTION_CELL) |

**Return** int 0 if succeed, -1 if failed.

### 3.2.58 LIB816x_setPlaybackDisplayLayout

**Description** Set playback screen layout mode

**Prototype** *int LIB816x_setPlaybackDisplayLayout(int mode, int nChannelIndex)*

**Arguments**

| | | |
|---|---|---|
| mode | [IN] | Screen layout mode |
| nChannelIndex | [IN] | N/A. |

**Return** int 1 fixed.

### 3.2.59 LIB816x_setPlaybackProperty_x

**Description** Set playback enable or speed

**Prototype** *int LIB816x_setPlaybackProperty_x(int cmd, int ch_bitmask, int value, void *pData)*

**Arguments**

| | | |
|---|---|---|
| cmd | [IN] | Channel enable or speed |

| | | | |
|---|---|---|---|
| | ch_bitmask | [IN] | Set channels bitmask |
| | value | [IN] | Speed when cmd is SET_SPEED |
| | pData | [IN] | N/A |

**Return** int 0 fixed

### 3.2.60 LIB816x_setPtzSerialInfo

**Description** Set PTZ serial configuration.

**Prototype** *int LIB816x_setPtzSerialInfo(int ptzDataBit, int ptzParityBit, int ptzStopBit, int ptzBaudRate)*

| | | | |
|---|---|---|---|
| **Arguments** | ptzDataBit | [IN] | Data bit |
| | ptzParityBit | [IN] | Parity bit |
| | ptzStopBit | [IN] | Stop bit |
| | ptzBaudRate | [IN] | Baud rate |

**Return** int 0 fixed

### 3.2.61 LIB816x_setRecChannel

**Description** Set record configuration by particular channel id.

**Prototype** *int LIB816x_setRecChannel(int chId, int enableRec, int eventMode, int addAudio, char* camName)*

| | | | |
|---|---|---|---|
| **Arguments** | chId | [IN] | Channel number |
| | enableRec | [IN] | Enable record |
| | eventMode | [IN] | Event mode |
| | addAudio | [IN] | Enable audio |
| | camName | [IN] | Camera title |

**Return** int 1 if succeed, 0 if failied

### 3.2.62 LIB816x_setRecDuration

**Description** Set pre-record enable and duration.

**Prototype** *int LIB816x_setRecDuration(int Ch,int bPrevRecEnable, int iPrevDuration)*

| | | | |
|---|---|---|---|
| **Arguments** | Ch | [IN] | Channel number |
| | bPrevRecEnable | [IN] | Enable pre-record |
| | iPrevDuration | [IN] | Duration of pre-record |

**Return** int 0 fixed

### 3.2.63 LIB816x_setRecordingType

**Description** Set type of using storage, recycle or once mode

**Prototype** *int LIB816x_setRecordingType(int iRecycle)*

| | | | |
|---|---|---|---|
| **Arguments** | iRecycle | [IN] | 0:recycle, 1:once |

**Return** int 1 fixed

### 3.2.64 LIB816x_setSensor

**Description** Set Sensor enable/disable and sensor type.

**Prototype** *void LIB816x_setSensor(int iSensorId,int iSensorEnable,int iSensorType)*

| | | | |
|---|---|---|---|
| **Arguments** | iSensorId | [IN] | Sensor index ( cf, MAX_SENSOR) |
| | iSensorEnable | [IN] | Enable flag |
| | iSensorType | [IN] | 0:NO(falling), 1:NC(Rising) |

**Return** void

### 3.2.65 LIB816x_setSpotChannel

**Description** Switch the channel Id for SDTV live bypass path

**Prototype** *int LIB816x_setSpotChannel(int chId)*

| | | | |
|---|---|---|---|
| **Arguments** | chId | [IN] | channel id to switch to |
| **Return** | int | 0 if succeed, 1 if failied. | |

### 3.2.66 LIB816x_setVideoCodecType

| | | | |
|---|---|---|---|
| **Description** | N/A. Set the channel Id for video codec. | | |
| **Prototype** | *int LIB816x_setVideoCodecType(int Ch,int iCodecType)* | | |
| **Arguments** | Ch | [IN] | Channel id |
| | iCodecType | [IN] | Codec type |
| **Return** | int | 0 fixed. | |

### 3.2.67 LIB816x_setVideoResolution

| | | | |
|---|---|---|---|
| **Description** | N/A. Set the channel Id for video codec. | | |
| **Prototype** | *int LIB816x_setVideoResolution(int Ch,int iResolution)* | | |
| **Arguments** | Ch | [IN] | Channel id |
| | iResolution | [IN] | Resolution type(D1, CIF, HALFD1 and etc) |
| **Return** | int | 0 if succeed, others failed | |

### 3.2.68 LIB816x_startCamProperty

| | | | |
|---|---|---|---|
| **Description** | Start PIP channel display | | |
| **Prototype** | *void LIB816x_startCamProperty(int selectedCh, int startX, int startY, int width, int height)* | | |
| **Arguments** | selectedCh | [IN] | Channel id |
| | startX | [IN] | Start X-position |
| | startY | [IN] | Start Y-position |
| | Width | [IN] | Width |
| | Height | [IN] | Height |
| **Return** | void | | |

### 3.2.69 LIB816x_startPlayback_x

**Description** start playback

**Prototype** *int LIB816x_startPlayback_x(int ch_bitmask, struct tm *ptm)*

**Arguments**

| | | |
|---|---|---|
| ch_bitmask | [IN] | Channel bitmask for play |
| ptm | [IN] | Start time |

**Return** int 0 if succeed, others failed

### 3.2.70 LIB816x_startRTSP

**Description** Start RTSP server thread.

**Prototype** *void LIB816x_startRTSP (void)*

**Arguments** none

**Return** Void 0 if succeed, others failed

### 3.2.71 LIB816x_startSystem

**Description** start dvr system

**Prototype** *int LIB816x_startSystem(int disp_main, int disp_sub, int bNtsc, int vmode)*

**Arguments**

| | | |
|---|---|---|
| disp_main | [IN] | Main display |
| disp_sub | [IN] | Sub display |
| bNtsc | [IN] | Is NTSC or PAL? |
| vmode | [IN] | N/A |

**Return** int 0 if succeed, others failed

### 3.2.72 LIB816x_stepPlayback_x

**Description** step play

**Prototype** *int LIB816x_stepPlayback_x(int ch_bitmask, int bReverse)*

**Arguments**

| | | |
|---|---|---|
| ch_bitmask | [IN] | Channel bitmask for play |
| bReverse | [IN] | 1:Reverse or 0:Forward |

**Return** int 0 if succeed, others failed

### 3.2.73 LIB816x_stopPlayback_x

**Description** Stop playback

**Prototype** *int LIB816x_stopPlayback_x(void)*

**Arguments** none

**Return** int 0 fixed

### 3.2.74 LIB816x_stopSystem

**Description** Stop dvr system

**Prototype** *int LIB816x_stopSystem(void)*

**Arguments** none

**Return** int 0 fixed

### 3.2.75 LIB816x_sys_info

**Description** Get hardware information. mac address and hw version

**Prototype** *int LIB816x_sys_info(char *mac0, char *mac1, char *hwver)*

**Arguments**

| | | |
|---|---|---|
| mac0 | [OUT] | Mac address 0 |
| mac1 | [OUT] | Mac address 1 |
| hwver | [OUT] | H/W version |

**Return** int 0 if succeed, -1 if failed

### 3.2.76 LIB816x_systemReboot

**Description** System reboot

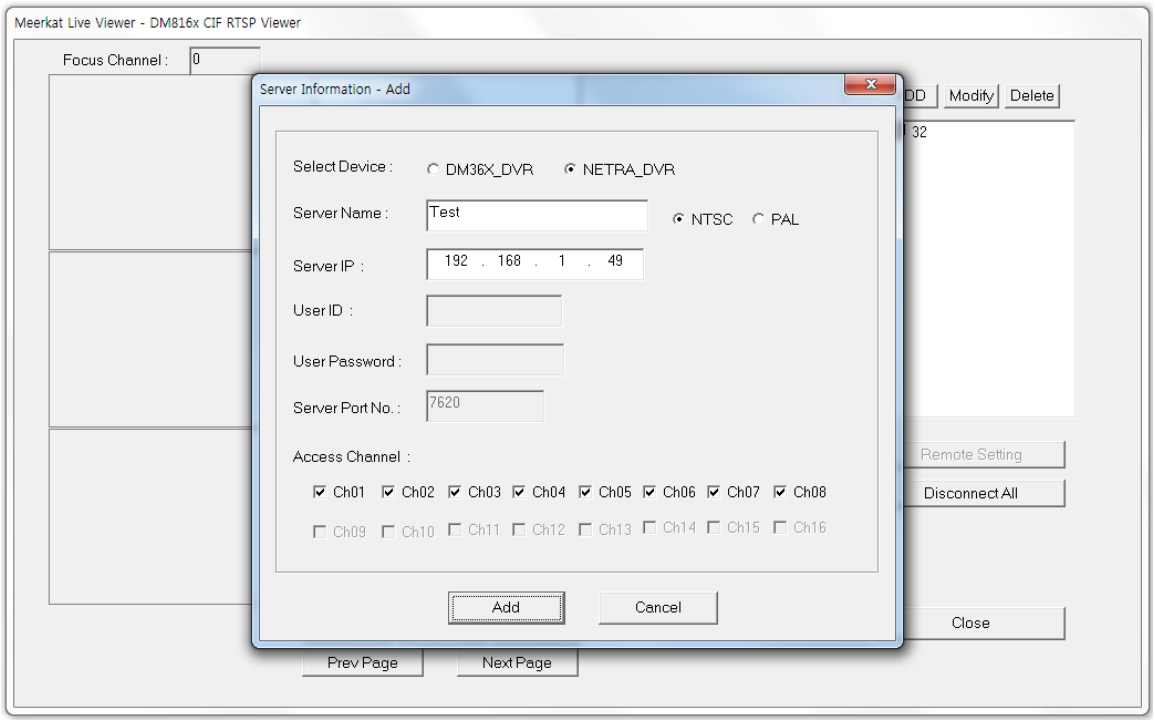
**Prototype** *void LIB816x_systemReboot(void)*

**Arguments** none

**Return** void

## 3.3 Netra VMS for RTSP

- Click [Add] button, enter the DVR name and IP address.




- You can see tree icon for DVR.

- Double-click using mouse or drag&drop the icon. This will enable the access for that particular channel from the DVR.