DVS-350

4~16 Channels Embedded/
Mobile Digital Video System

User Manual

Copyright and Disclaims

Acknowledgements

Intel and Pentium are trademarks of Intel Corporation.
Microsoft Windows and MS-DOS are registered trademarks of Microsoft Corp.
All other product names or trademarks are properties of their respective owners.

Part No. 2002S35000
May 2006 1st Edition
Printed in Taiwan

Product Warranty (1 year)

Advantech warrants to you, the original purchaser, that each of its products will be free from defects in materials and workmanship for one year from the date of purchase.

This warranty does not apply to any products which have been repaired or altered by persons other than repair personnel authorized by Advantech, or which have been subject to misuse, abuse, accident or improper installation. Advantech assumes no liability under the terms of this warranty as a consequence of such events.

Because of Advantech's high quality-control standards and rigorous testing, most of our customers never need to use our repair service. If an Advantech product is defective, it will be repaired or replaced at no charge during the warranty period. For out-of-warranty repairs, you will be billed according to the cost of replacement materials, service time and freight. Please consult your dealer for more details.

If you think you have a defective product, follow these steps:
1. Collect all the information about the problem encountered. (For example, CPU speed, Advantech products used, other hardware and software used, etc.) Note anything abnormal and list any onscreen messages you get when the problem occurs.

2. Call your dealer and describe the problem. Please have your manual, product, and any helpful information readily available.

3. If your product is diagnosed as defective, obtain an RMA (return merchandize authorization) number from your dealer. This allows us to process your return more quickly.

4. Carefully pack the defective product, a fully-completed Repair and Replacement Order Card and a photocopy proof of purchase date (such as your sales receipt) in a shippable container. A product returned without proof of the purchase date is not eligible for warranty service.

5. Write the RMA number visibly on the outside of the package and
ship it prepaid to your dealer.

Declaration of Conformity

CE

This product has passed the CE test for environmental specifications when shielded cables are used for external wiring. We recommend the use of shielded cables. This kind of cable is available from Advantech. Please contact your local supplier for ordering information.

This product has passed the CE test for environmental specifications. Test conditions for passing included the equipment being operated within an industrial enclosure. In order to protect the product from being damaged by ESD (Electrostatic Discharge) and EMI leakage, we strongly recommend the use of CE-compliant industrial enclosure products.

FCC Class A

Note: This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at his own expense.

Technical Support and Assistance

Step 1. Visit the Advantech web site at www.advantech.com/support
where you can find the latest information about the product. Contact your distributor, sales representative, or Advantech's customer service center for technical support if you need additional assistance.

Please have the following information ready before you call:
- Product name and serial number
- Description of your peripheral attachments
- Description of your software (operating system, version, application software, etc.)
- A complete description of the problem

- The exact wording of any error messages

DVS-350 Series Model

There are 6 sub-models in DVS-350 series listed below:

Part Number	Video Channel	CPU	L2 Cache
DVS-350-18M0	4~16 CH	Intel ULV Celeron® M 600MHz	512KB
DVS-350-18S4	4~16 CH	Intel LV Pentium® M 1.4GHz	2MB
DVS-350-25M0	4~16 CH	Intel ULV Celeron® M 600MHz	512KB
DVS-350-25S4	4~16 CH	Intel LV Pentium® M 1.4GHz	2MB
DVS-350-35M0 *	4~16 CH	Intel ULV Celeron® M 600MHz	512KB
DVS-350-35S4 *	4~16 CH	Intel LV Pentium® M 1.4GHz	2MB
* The swappable HDD tray of DVS-350-35M0/DVS-350-35S4 just support HITACHI and Seagate HDD only.			

Table 0.1 DVS-350 Model List

Packing List

Before installing your board, make sure that the following materials have been received:

Part Number	Description	Quantity
2013000000	1 year warranty card	1
2062S35000	Driver CD	1
1700001394	DC Jack with 2-pin pluggable terminal block	1
1652000209	Pluggable terminal block for DI and DO	1
1700060202	Y cable of KB and PS/2 Mouse	1
1700001618	Video Cable (D-sub 15P to BNC)	2

* DVS-350-35M0/DVS-350-35S4 also has a key for swappable HDD tray lock door.

If any of these items are missing or damaged, contact your distributor or sales representative immediately.

Safety Instructions

1. Please read these safety instructions carefully.

2. Please keep this User Manual for later reference.

3. Please disconnect this equipment from power outlet before cleaning. Don‘t use liquid or sprayed detergent for cleaning. Use moisture sheet or clothe for cleaning.

4. For pluggable equipment, the socket-outlet shall near the equipment and shall be easily accessible.

5. Please keep this equipment from humidity.

6. Lay this equipment on a reliable surface when install. A drop or fall could cause injury.

7. Do not leave this equipment in an uncontrolled environment; storage temperatures above 60ºC may damage the equipment.

8. The openings on the enclosure are for air convection hence protecting the equipment from overheating. DO NOT COVER THE OPENINGS.

9. Make sure the voltage of the power source when connecting the equipment to the power outlet.

10. Place the power cord such a way that people cannot step on it. Do not place anything over the power cord. The power cord must be rated for the product and for the voltage and current marked on the product’s electrical ratings label. The voltage and current rating of the cord should be greater than the voltage and current rating marked on the product.

11. All cautions and warnings on the equipment should be noted.

12. If the equipment is not used for long time, disconnect the equipment from mains to avoid being damaged by transient over-voltage.

13. Never pour any liquid into ventilation openings; this could cause fire or electrical shock.

14. Never open the equipment. For safety reasons, only qualified service personnel should open the equipment.

15. If one of the following situations arise, get the equipment checked by service personnel:

a. The Power cord or plug is damaged.

b. Liquid has penetrated the equipment.

c. The equipment has been exposed to moisture.

d. The equipment has not worked well or you can not get it work according to user‘s manual.

e. The equipment has been dropped and damaged.

f. The equipment has obvious signs of breakage

16. This device complies with Part 15 of the FCC rules. Operation is subject to the following two conditions:

a. this device may not cause harmful interference, and

b. this device must accept any interference received, including interference that may cause undesired operation.

CAUTION!

THIS COMPUTER IS PROVIDED WITH A BATTERY-POWERED REAL-TIME CLOCK CIRCUIT. THERE IS A DANGER OF EXPLOSION IF BATTERY IS INCORRECTLY REPLACED. REPLACE ONLY WITH SAME OR EQUIVLENT TYPE RECOMMENDED BY THE MANUFACTURE. DISCARD USED BATTERIES ACCORDING TO THE MANUFACTURER’S INSTRUCTIONS.

Wichtige Sicherheishinweise

1. Bitte lesen sie Sich diese Hinweise sorgfältig durch.

2. Heben Sie diese Anleitung für den späteren Gebrauch auf.

3. Vor jedem Reinigen ist das Gerät vom Stromnetz zu trennen. Verwenden Sie Keine Flüssig-oder Aerosolreiniger. Am besten dient ein angefeuchtetes Tuch zur Reinigung.

4. Die NetzanschluBsteckdose soll nahe dem Gerät angebracht und leicht zugänglich sein.

5. Das Gerät ist vor Feuchtigkeit zu schützen.

6. Bei der Aufstellung des Gerätes ist auf sicheren Stand zu achten. Ein Kippen oder Fallen könnte Verletzungen hervorrufen.

7. Die Belüftungsöffnungen dienen zur Luftzirkulation die das Gerät vor überhitzung schützt. Sorgen Sie dafür, daB diese Öffnungen nicht abgedeckt werden.

8. Beachten Sie beim. AnschluB an das Stromnetz die AnschluBwerte.

9. Verlegen Sie die NetzanschluBleitung so, daB niemand darüber fallen kann. Es sollte auch nichts auf der Leitung abgestellt werden.

10. Alle Hinweise und Warnungen die sich am Geräten befinden sind zu beachten.

11. Wird das Gerät über einen längeren Zeitraum nicht benutzt, sollten Sie es vom Stromnetz trennen. Somit wird im Falle einer Überspannung eine Beschädigung vermieden.

12. Durch die Lüftungsöffnungen dürfen niemals Gegenstände oder Flüssigkeiten in das Gerät gelangen. Dies könnte einen Brand bzw. Elektrischen Schlag auslösen.

13. Öffnen Sie niemals das Gerät. Das Gerät darf aus Gründen der elektrischen Sicherheit nur von authorisiertem Servicepersonal geöffnet werden.

14. Wenn folgende Situationen auftreten ist das Gerät vom Stromnetz zu trennen und von einer qualifizierten Servicestelle zu überprüfen:

   a. Netzkabel oder Netzstecker sind beschädigt.
   b. Flüssigkeit ist in das Gerät eingedrungen.

c. Das Gerät war Feuchtigkeit ausgesetzt.

d. Wenn das Gerät nicht der Bedienungsanleitung entsprechend funktioniert oder Sie mit Hilfe dieser Anleitung keine Verbesserung erzielen.

e. Das Gerät ist gefallen und/oder das Gehäuse ist beschädigt.

f. Wenn das Gerät deutliche Anzeichen eines Defektes aufweist.

15. VOSICHT: Explisionsgefahr bei unsachgemaben Austausch der Batterie. Ersatz nur durch densellben order einem vom Hersteller empfohlene-mahnlichen Typ. Entsorgung gebrauchter Batterien navh Angaben des Herstellers.

16. ACHTUNG: Es besteht die Explosionsgefahr, falls die Batterie auf nicht fach-männische Weise gewechselt wird. Verfangen Sie die Batterie nur gleicher oder entsprechender Type, wie vom Hersteller empfohlen. Entsorgen Sie Batterien nach Anweisung des Herstell-ers.

Der arbeitsplatzbezogene Schalldruckpegel nach DIN 45 635 Teil 1000 beträgt 70dB(A) oder weiger.

Haftungsausschluss: Die Bedienungsanleitungen wurden entsprechend der IEC-704-1 erstellt. Advantech lehnt jegliche Verantwortung für die Richtigkeit der in diesem Zusammenhang getätigten Aussagen ab.

Safety Precaution - Static Electricity

Follow these simple precautions to protect yourself from harm and the products from damage.

1. To avoid electrical shock, always disconnect the power from your PC chassis before you work on it. Don't touch any components on the CPU card or other cards while the PC is on.

2. Disconnect power before making any configuration changes. The sudden rush of power as you connect a jumper or install a card may damage sensitive electronic components.

CHAPTER 1

Overview

This chapter gives background information on the DVS-350 series. It shows you the DVS-350 overview and specifications.

Sections include:

- Introduction
- Hardware Specifications
- DVS-350 Series Model and Packing List
- Chassis Dimension

Chapter 1 Overview

1.1 Introduction

The DVS-350 Digital Video System combines rich video capture functions and other industrial features into a rugged, compact metal chassis for digital video applications. The fanless operation provides high reliability when deployed in space constrained environments. All electronics are protected in a sealed anti-vibration anti-dust housing, making the DVS-350 ideal for harsh environment applications. Its X86 architecture offers an open platform for easy application development or inclusion of other applications like vehicle data recorders or global positioning systems (GPS).

1.2 Features

1.2.1 High Video Capacity and Easy Integration

・ DVS-350 uses 4 Conexant Fusion 878A video capture chips which are certificated and commonly used in digital video recording market. For more information about the BT878 chip, please visit http://www.conexant.com/products/entry.jsp?id=272

DVS-350 can support up to D1 resolution with total frame rate 120fps/NTSC or 100fps/PAL. Each Conexant Fusion 878A can switch 4 video inputs. With this share frame technology, DVS-350 can run up to 16 channel video inputs.

Also, DVS-350 provides users the software development kit (SDK) for Windows-based environment with sample program and its complete C++ source codes, which will speed the time of system integration and save your money.

Moreover, DVS-350 comes with PowerView DVR application software (Trial Version, Valid for 60 days only.) Users can easily evaluate and experience the performance of DVS-350 via PowerView. If interested, users can contact with Advantech's sale representative for further customization and speedy integration by project-based.

1.2.2 Robust Casting Construction

・ Fan-less operation in Aluminum sealed construction

・ A special cushioned design that absorbs vibration to ensure maximum reliability under harsh conditions

1.2.3 Compact Size

・ With its maximum mounting height of 45.8 mm (DVS-350-18M0), the DVS-350 can be used under space critical installation conditions

1.2.4 Scalable Performance with low power consumption

・ Scalable Low Voltage and Ultra Low Voltage Pentium M class processor system to bring high computing performance with low power consumption

1.2.5 Optimized Integration

・ Few Parts, Easy Integration, Easy Maintenance to reduce investment
・ Systems are supplied ready to run
・ Long life cycle support for product continuity

1.2.6 Wide Range of Power Source

Wide range of DC 9V~30V power source offers flexibility of power input for various automation environments.

1.2.7 Options for Expansion

DVS-350 provides 5 Isolated DI, 2 Relay DO, 1 Remote SW control, COM1: RS-232, COM2: RS-485/RS-422 and 4xUSB 2.0 ports which can fit most of application scenario. If users need more DI, DO or COM ports, Advantech provides the following option for easy expansion:

RS-232/422/485 expansion option	
Product Name	Description
EDG-4504	4-port RS-232/422/485 to Ethernet data gateway
ADAM-4570	2-port RS-232/422/485 to Ethernet data gateway
ADAM-4571	1-port RS-232/422/485 to Ethernet data gateway
ADAM-4570L	2-port RS-232 to Ethernet data gateway
ADAM-4570L	1-port RS-232 to Ethernet data gateway
ADAM-4561	1-port isolated USB to RS-232/422/485 Converter

Ethernet expansion option	
Product Name	Description
ADAM-6510	4-port industrial 10 Mbps Ethernet hub
ADAM-6520	5-port industrial 10/100 Mbps Ethernet hub
ADAM-6521	5-port industrial 10/100 Mbps Ethernet switch with fiber port
UNO-2058	GX1-300 Universal Network Controller with GPS/GPRS Communication

Digital I/O expansion option	
Product Name	Description

ADAM-4501	Ethernet-enabled Communication Controller with 8 Digital I/O
ADAM-4052	Isolated Digital Input Module
ADAM-4053	16-channel Digital Input Module
ADAM-4055	16-channel Isolated Digital I/O Module with LED and Modbus
ADAM-4056S	12-channel Sink Type Isolated Digital Output Module
ADAM-4056S0	12-channel Source Type Isolated Digital Output Module
ADAM-4060	4-channel Relay Output Module
ADAM-4068	8-channel Relay Output Module with Modbus and LED

For further information of above options, please visit http://www.advantech.com

1.2 Hardware Specification

1.2.1 Processor Core Logic System

CPU Type: Intel® Ultra Low Voltage Celeron® M or Intel® Pentium® M Low Voltage Processor, µFC-BGA 479 Package:

Model Name	Part Number	Intel® CPU Type	Intel® Chipset
DVS-350C	DVS-350-18M0	Intel ULV Celeron® M 600MHz	Intel® 852GM / ICH4
	DVS-350-18S4	Intel LV Pentium® M 1.4GHz	Intel® 852GM / ICH4
DVS-350M	DVS-350-25M0	Intel ULV Celeron® M 600MHz	Intel® 852GM / ICH4
	DVS-350-25S4	Intel LV Pentium® M 1.4GHz	Intel® 852GM / ICH4
DVS-350F	DVS-350-35M0	Intel ULV Celeron® M 600MHz	Intel® 852GM / ICH4
	DVS-350-35S4	Intel LV Pentium® M 1.4GHz	Intel® 852GM / ICH4

BIOS: 4Mbit Flash BIOS, supports Plug & Play, APM 1.2

System Memory

- One 200 pin SO-DIMM sockets, support ECC DDR SDRAM Up to 1GB, DDR200/266/333 DRAM

1.2.2 Display

Chipset

・ Integrated graphics built-in Intel® 852GM GMCH, utilizing Intel® Extreme Graphics 2 technology

Display Memory

・ Dynamic video memory allocation up to 32 MB

Display Interface support

- VGA and TV-out (BNC) Interface

1.2.3 Ethernet

- Ethernet Controller: Intel® 82551QM Fast Ethernet Multifunction PCI/CardBus Controller
- Speed: 10/100MBps, IEEE 802.3u (100 BASE-T) protocol compatible

1.2.4 Other

- Watchdog Timer: 255 levels timer interval, setup by software
- Serial Port: One RS-232 port (COM1) and One RS-485/422 port (COM2)
- Keyboard/Mouse: One PS/2 Port to support PS/2 Mouse and PS/2 Keyboard
- USB: 4 USB 2.0 compliant universal serial bus port

1.2.5 Storage

- Supports a drive bay space for 1.8”, 2.5’’ and 3.5” HDD
- Supports a Compact Flash socket for Type I/II Compact Flash disk

1.2.6 Mechanical

- Construction: Aluminum housing
- Mounting: DIN-rail mounting, Desk/wall mounting
- Dimension (W x H x D):

DVS-350C: 232.0 x 138.4 x 45.0 mm (9.13” x 5.45” x 1.8”); Weight: 2.0KG

DVS-350M: 232.0 x 138.4 x 65.0 mm (9.13” x 5.45” x 2.56”); Weight: 2.3KG

DVS-350F: 232.0 x 138.4 x 65.0 mm (9.13” x 5.45” x 3.54”); Weight: 2.9KG

1.2.7 Power Supply

- Maximum Output Rating 45 W
- Inside Fuse Rating 7 A
- Input Voltage: 9 ~ 30V DC,

Typical:

9 VDC @ 5.4A,

19 VDC @ 3.42 A,

1.2.8 Environment Specifications

- Operating Temperature: -10 to 45° C

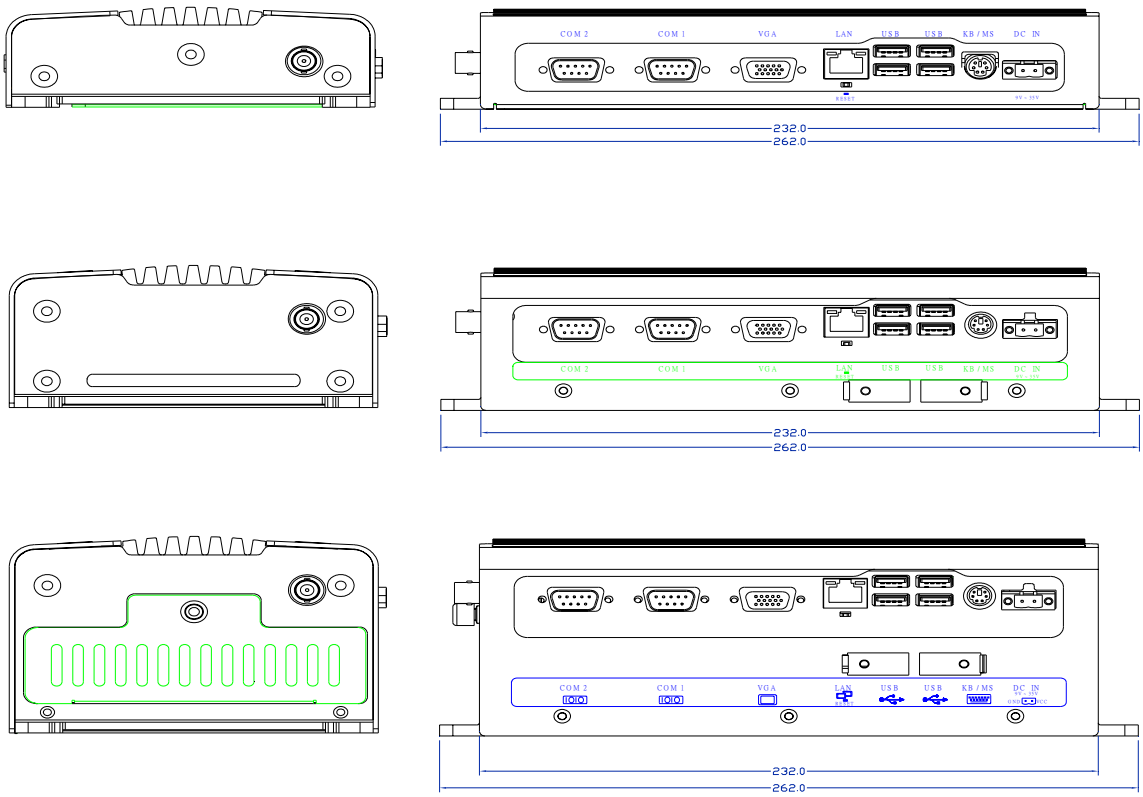
Warning: *Please do not keep DVS-350 working in a closed environment. The temperature in a closed environment might get higher over the operation temperature.*

- Relative humidity 95 % @ 40 ° C (non-condensing)
- Vibration loading during operation: 1Grms, IEC 68-2-64, random, 5~500Hz, 1 Oct./min, 1hr/axis.
- Shock during operation: 20Grms, IEC 60068-2-27, half sine, 11 ms duration

1.2.9 Certificate Approved

- EMC Approved: CE, FCC Class A
- Safety Approved: UL

1.3 Dimensions Diagram



Unit: mm

CHAPTER 2

Hardware Functionality

This chapter shows how to set up the DVS-350 hardware functions, including connecting peripherals, switches and indicators.

Sections include:

- Introduction of External I/O Connectors
- Front plate external I/O Connectors
- Power Connector
- LED Indicators
- Video Input Connectors
- Isolated Digital I/O
- Audio Connectors
- Rear plate external I/O Connectors
- COM1 Connector
- COM2 Connector
- Ethernet Connector
- Reset Button
- PS2 Keyboard/Mouse Connector

Chapter 2 Hardware Functionality

2.1 Introduction of DVS-350 External I/O Connectors

The following two figures show the external I/O connectors on DVS-350. The following sections give you detailed information about the function of each I/O connector.

Figure 2.1: DVS-350 front metal face plate external I/O connectors

Figure 2.2: DVS-350 rear metal face plate I/O connectors

2.2 DVS-350 front plate external I/O connectors

2.2.1 Power ON/OFF Button

The DVS-350 comes with a Power On/Off button, that Support dual function Soft Power - On/Off (Instant off or Delay 4 Second), and Suspend.

2.2.2 LED Indicators

There are two LEDs on the DVS-350 front face plate for indicating system status: PWR LED is for power status and flash in green color; and HDD LED is for hard disk and compact flash disk status, which flash in red color.

2.2.3 Video Input Connectors

Figure 2.3: Video Input Connectors

The DVS-350 can support up to 16 channel video inputs with 2 customization video cables. The pin definition of video connector and video cable is following:

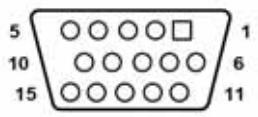


Table 2.1 Video Input Connector			
Pin	Definition	Pin	Definition
1	Channel 1 /9	9	Channel 5 /13
2	Ground	10	Ground
3	Channel 2 /10	11	Channel 6 /14
4	Ground	12	Ground
5	Channel 3 /11	13	Channel 7 /15
6	Ground	14	Ground
7	Channel 4 /12	15	Channel 8 /16
8	Ground	N/A	N/A

DVS-350 has a high integration main board with 4 Conexant Fusion 878A video chips. Each 878A chip can run with 30 frames per second at NTSC or 25 fps at PAL up to D1 resolution. Also, each 878A chip can accept 4 channel video input to share the total 30/25 frames. Following table is the video performance of 1 Conexant Fusion 878A chip with enabling share frame function.

Table 2.2 Video performance with share frame function				
Input channel	1 channel	2 channel	3 channel	4 channel
Share frame per second / NTSC	30 fps	15 fps	10 fps	7.5 fps
Share frame per second / PAL	25 fps	12.5 fps	8 fps	6 fps

Table 2.3 DVS-350 Video Channel Arrangement		
Channel Number	878A Video Chip Number	Video Input (Chip-Switch)
Channel 1	Chip 1	1-1
Channel 2	Chip 2	2-1
Channel 3	Chip 3	3-1
Channel 4	Chip 4	4-1
Channel 5	Chip 1	1-2
Channel 6	Chip 2	2-2
Channel 7	Chip 3	3-2
Channel 8	Chip 4	4-2
Channel 9	Chip 1	1-3
Channel 10	Chip 2	2-3
Channel 11	Chip 3	3-3
Channel 12	Chip 4	4-3
Channel 13	Chip 1	1-4
Channel 14	Chip 2	2-4
Channel 15	Chip 3	3-4
Channel 16	Chip 4	4-4

If user would like to run DVS-350 with 4 channel only, user should connect the 4 channel inputs to channel 1 to channel 4. Then each channel can run at 30/25 frame per second.

If user would like to run DVS-350 with 8 channel application, user should connect the 8 channel inputs to channel 1 to channel 8 with 15/12.5 fps per channel.

If user would like to run DVS-350 with 12 channel application, user should connect the 12 channel inputs to channel 1 to channel 12 with 10/8 fps per channel.

If user would like to run DVS-350 with 16 channel application, user should connect the 16 channel inputs to channel 1 to channel 16 with 7.5/6 fps per channel.

Also, user can set different 878A chip with different switch. For example, user can set the chip1 to 1 input only, the chip2 to 3 inputs, the chip3 to 2 inputs, and the chip4 to 4 inputs. Then, the channel 1 can run at 30/25 fps. The channel 2, 6 and 10 can run at 10/8 fps. The channel 3 and 7 can run at 15/12.5 fps. The channel 4, 8, 12 and 16 can run at 7.5/6 fps. The channel 5, 9, 11, 13, 14 and 15 will not accept any video signal with above setting. User can set the video channel setting to meet the specific application situation.

For the DVS-350 series with Intel ULV Celeron® M 600MHz, user just can run it up to 8 video inputs at 15/12.5fps.

Note: *Above information are the characteristics of Conexant Fusion 878A video chip and DVS-350 series. The actual frame rate and performance will depend on the video input setting, compression codec, the CPU type, the DRAM capacity/speed, the video content (Moving or still), the brightness and etc.*

2.2.4 Isolated DI, Relay DO and Remote SW control

Figure 2.4: Isolated DI, Relay DO and Remote SW control

Isolated DI Channel: 5 fully independent isolated channels

Digital Input Level:

Logic level 0: 0V~3V

Logic level 1: 10V~30V

Isolation Voltage 3,750V RMS

Relay DO Channel: 2 channels, 2 Form C

Contact Rating:

AC: 125V at 0.5A

DC: 30V at 2A, 110V at 0.3A

Breakdown Voltage: 500 V AC (50/60Hz)

Relay on Time (Typical): 2ms

Relay off Time (Typical): 2ms

Insulation Resistance: 1,000MΩ minimum at 500V DC

Table 2.4 Isolated DI, Relay DO and Remote SW control

Pin	Definition	Pin	Definition
1	DO_1+	9	DI_3+
2	DO_1-	10	DI_3-
3	DO_2+	11	DI_4+
4	DO_2-	12	DI_4-
5	DI_1+	13	DI_5+
6	DI_1-	14	DI_5-
7	DI_2+	15	PSW+
8	DI_2-	16	PSW-

2.2.5 MIC-in, Line-out and LED Indicators

2.3 DVS-350 rear plate external I/O connectors

2.3.1 Power Input Connector

The DVS-350 comes with a Phoenix connector that carries 9~30 VDC external power input.

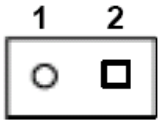


Table 2.5 Power connector pin assignments			
Pin	Definition	Pin	Definition
1	Ground	2	+9~30VDC

2.3.2 COM1 Connector

The DVS-350 provides a D-sub 9-pin connector, which offers one standard RS-232 serial communication interface port of COM1.

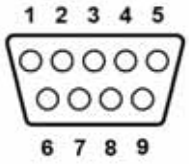


Table 2.6 COM1 standard serial port pin assignments			
Pin	Definition	Pin	Definition
1	DCD	6	DSR
2	RxD	7	RTS
3	TxD	8	CTS
4	DTR	9	RI
5	GND	N/A	N/A

2.3.3 COM2 Connector

The DVS-350 provides a D-sub 9-pin connector, which offers one RS-485 serial communication interface port for COM2. The default setting of COM2 is RS-485.

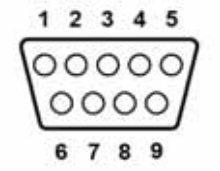


Table 2.7 COM2 standard serial port pin assignments	
Pin	RS-485
1	DATA-
2	DATA+
3	No Connection
4	No Connection
5	GND
6	No Connection
7	No Connection
8	No Connection
9	No Connection

2.3.4 Ethernet Connector (LAN)

The DVS-350 is equipped with an Intel 82551QM Fast Ethernet controller that is fully compliant with IEEE 802.3u 10/100Base-T CSMA/CD standards. The Ethernet port provides a standard RJ-45 jack connector with LED indicators on the front side to show its Active/Link status (Green LED) and Speed status (white LED).

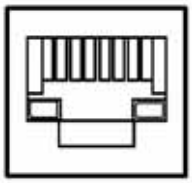


Table 2.8 RJ-45 Connector pin assignments	
Pin	10/100BaseT Signal Name
1	XMT+
2	XMT-
3	RCV+
4	No Connection
5	No Connection
6	RCV-
7	No Connection
8	No Connection

2.3.5 Reset Button

Press the "Reset" button to activate the reset function.

2.3.6 PS2 Keyboard/Mouse Connector

The DVS-350 provides a PS/2 keyboard/mouse connector. A 6-pin mini-DIN connector is located on the rear face plate. It comes with an external Y cable to convert from the 6-pin mini-DIN connector to PS/2 keyboard and PS/2 mouse connection.

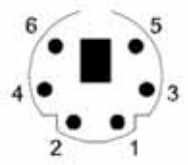


Table 2.9 PS2 Keyboard/Mouse Connector	
Pin	Definition
1	PS2_KBDAT
2	PS2_MSDAT
3	GND
4	VCC
5	PS2_KBCLK
6	PS2_MSCLK

2.3.7 VGA Connector

The DVS-350 provides a high resolution VGA interface by a D-sub 15-pin connector to support a VGA CRT monitor. It supports VGA and VESA, up to 1600 x 1200 @85-Hz resolution and up to 32 MB shared memory.

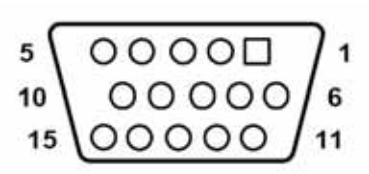


Table 2.10 VGA Connector			
Pin	Definition	Pin	Definition
1	Red	9	No Connection
2	Green	10	GND
3	Blue	11	No Connection
4	No Connection	12	No Connection
5	GND	13	H-SYNC
6	GND	14	V-SYNC
7	GND	15	No Connection
8	GND	N/A	N/A

2.3.8 USB Connector

The DVS-350 provides 4 connectors for USB interface, which gives complete Plug & Play and hot swapping for up to 127 external devices. The USB interface complies with USB UHCI, Rev. 2.0 compliant. The USB interface can be disabled in the system BIOS setup.

The USB connector is used for connecting any device that conforms to the USB interface. Many recent digital devices conform to this standard. The USB interface supports Plug and Play, which enables you to connect or disconnect a device whenever you want, without turning off the computer.

Table 2.11 USB Connector			
Pin	Definition	Pin	Definition
1	VCC	3	USB_P+
2	USB_P-	4	GND

CHAPTER 3

Hardware Installation and Upgrade

This chapter introduces how to initialize the DVS-350.

Sections include:

- Installing the DDR SDRAM Memory Module
- Inserting a Compact Flash Card
- Installing the 2.5” Hard Disk Drive (HDD)
- Connecting Power

Chapter 3 Hardware Installation and Upgrade

3.1 Jumpers and Connectors

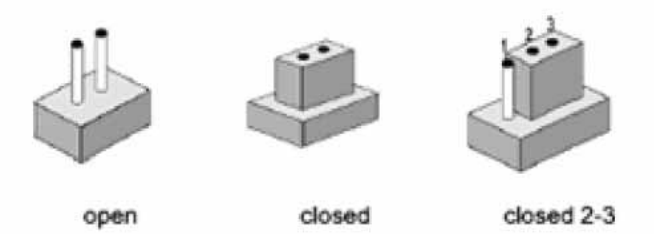
The DVS-350 Embedded/Mobile Video System consists of a PC-based computer that is housed in a aluminum top cover, a metal bottom case with accessed bottom cover and front with rear metal face plate. The HDD, SDRAM, are accessible by removing the accessed bottom cover. Any maintenance or hardware upgrades can be easily completed after removing the top cover, front and rear plates.

Warning!

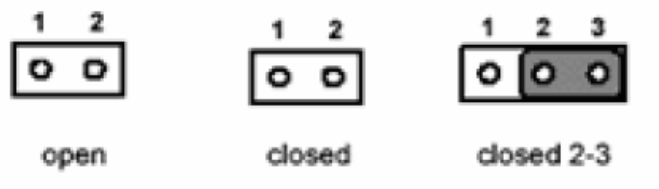
Do not remove any mechanical parts, such as the top cover, bottom cover and front with rear face plate until you have verified that no power is flowing within the embedded box computer. Power must be switched off and the power cord must be unplugged. Every time you service the embedded box computer, you should be aware of this.

3.2 Setting jumpers

You can configure your DVS-350 to match the needs of your application by setting jumpers. A jumper is the simplest kind of electrical switch. It consists of two metal pins and a small metal clip (often protected by a plastic cover) that slides over the pins to connect them. To “close” a jumper, you connect the pins with the clip. To “open” a jumper you remove the clip. Sometimes a jumper will have three pins, labeled 1, 2, and 3. In this case, you would connect either pins 1 and 2 or pins 2 and 3.



The jumper settings are schematically depicted in this manual as follows:



A pair of needle-nose pliers may be helpful when working with jumpers. If you have any doubts about the best hardware configuration for your application, contact your local distributor or sales representative before you make any changes.

3.3 COM2 RS-485/232 Jumper setting (JP2)

The COM2 port located on front metal face plate of DVS-350 unit which can be configured to operate in RS-485 (default setting) or RS-422 mode by setting up the Jumper Pins of JP2 located on internal motherboard of DVS-350 unit.

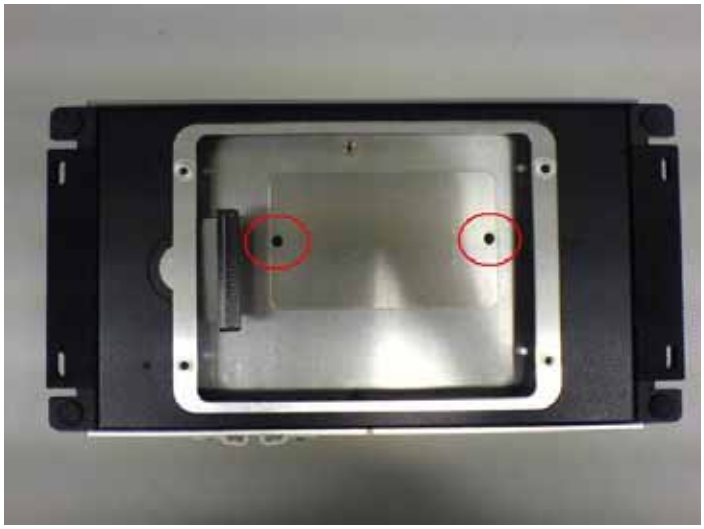
Table 3.1: JP2: COM2 RS-485/232 selection	
Setting	Function
JP2 (1-2 closed), (9-11 closed), (10-12 closed), (15-17 closed), (16-18 closed)	RS-485
JP2 (5-6 closed), (7-9 closed), (8-10 closed), (13-15 closed), (14-16 closed)	RS-232

3.4 Installing the DDR SDRAM Memory Module

The DVS-350 provides one 200-pin SODIMM (Small Outline Dual Inline Memory Module) socket and supports 2.5V DDR SDRAM. You can install from 64 MB up to 1 GB of DDR SDRAM memory. The procedure of installing a DDR SDRAM SODIMM into the DVS-350 is detailed below, please follow these steps carefully.

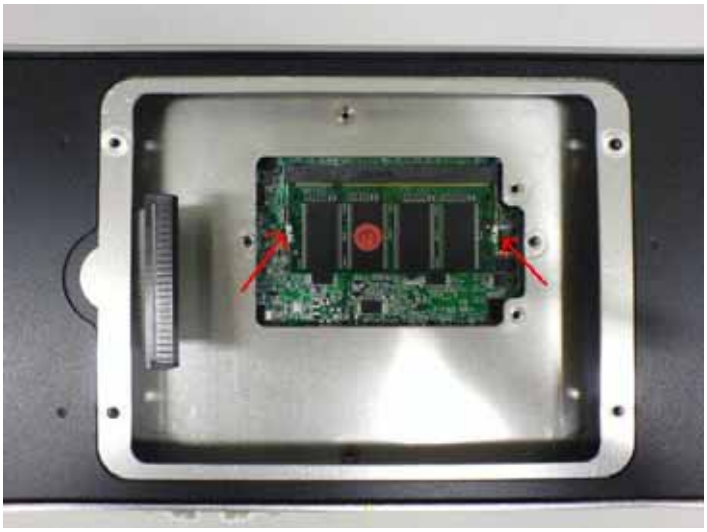
1. Remove the power line first.
2. Unscrew the screws from the bottom cover of the DVS-350.

3. Remove the bottom cover.
4. Unscrew the screws from the bottom cover of DDR SODIMM SDRAM



Notice: For DVS-350-35M0 and DVS-350-35S4, it is necessary to remove the swappable HDD tray by the key in accessory box first before unscrewing the screws from the bottom cover of DDR SODIMM SDRAM

4. Plug and Push a DDR SODIMM SDRAM (200pin) into a socket on board.
5. Press down the DRAM module and you will hear the “Click”.



6. Screw back the bottom cover of DDR SODIMM SDRAM with two screws and the bottom cover with four screws.

3.5 Installing the 2.5" Hard Disk Drive (HDD)

You can attach one enhanced Integrated Device Electronics (IDE) hard disk drive to the DVS-350's internal controller which uses a PCI localbus interface. The advanced IDE controller supports faster data transfer and allows the IDE hard drive to exceed 528 MB.

For the automotive application and harsh(high temperature) environment installation, we STRONGLY suggest users to use the wide-range 2.5" hard disk driver designed and manufactured specifically for these application. As we know, **Seagate EE25 Series** hard drive and **Fujitsu MHT2020AC** hard drive can meet higher operating temperature (–30º to 85ºC) and vibration tolerance (Up to 2.0 Gs). For more information about the wide-range hard drives, we suggest you **Google** the "Seagate EE25 Series" and "Fujitsu MHT2020AC" on web. You should find some useful information from the internet.

The following are instructions for installation:

1. Remove the power line first.
2. Unscrew the screws on HDD mounting bottom cover of the DVS-350

3. Remove the HDD mounting bottom cover of the DVS-350.

4. Fixing your HDD to HDD mounting bottom cover by using the 4 screws stored in accessories box. Make sure the PCB side down and the label side up.

5. Connect the IDE flat cable to the connector on the hard disk.

6. Make sure the tight connection between the flat cable and the hard disc drive.
7. Screw back the bottom cover with the four screws.

3.6 Inserting and Removing a Compact Flash Card

The procedure of installing a Compact Flash card into the DVS-350 is detailed below, please follow these steps carefully:

1. Remove the power line first.
2. Unscrew the two screws from the CF Door located on rear face plate of the DVS-350 embedded box computer.

3. Remove the CF cover.
4. Match the size of the slide on both sides of CF card first before inserting the CF card.

5. Insert the Compact Flash card slightly into the CF socket.

6. DO NOT push the CF card too hard when the slide is not matching. It might cause damage to the CF card and the socket. When it's not easy to slide the CF card into the socket, you need to flip the CF card over and try again.

7. Use a screw tool to push the CF card inward to make sure a tight connection between CF card and socket.

Warning: Be careful not to touch and push any component on the PCB board.
Notice: The Compact Flash socket is allocated as Secondary IDE Master (IDE-1).

3.7 Connecting Power

Connect the DVS-350 to a 12~24V DC power source. The power source can either be from a power adapter or an in-house power source.

Warning: For the automotive application, the power voltage is very unstable during the key ignition and might be lower to 7V DC. Please make sure the system is not under booting or operating. The unstable power source might cause the shot-down to the system and the data loss.

CHAPTER 4

Award BIOS Setup

Chapter 4 Award BIOS Setup

4.1 Introduction

Award’s BIOS ROM has a built-in setup program that allows users to modify the basic system configuration. This type of information is stored in battery-backed memory (CMOS RAM) so that it retains the setup information when the power is turned off.

4.1.1 CMOS RAM Auto-backup and Restore

The CMOS RAM is powered by an onboard button cell battery. When you finish BIOS setup, the data in CMOS RAM will be automatically backed up to Flash ROM. If operation in harsh industrial environments causes a soft error, BIOS will recheck the data in CMOS RAM and automatically restore the original data in Flash ROM to CMOS RAM for booting.

Note: If you intend to change the CMOS setting without restoring the previous backup, you have to click on "DEL" within two seconds of the "CMOS checksum error..." display screen message appearing. Then enter the "Setup" screen to modify the data. If the "CMOS checksum error..."message appears again and again, please check to see if you need to replace the battery in your system.

4.2 Entering Setup

Turn on the computer and check for the “patch code”. If there is a number assigned to the patch code, it means that the BIOS supports your CPU. If there is no number assigned to the patch code, please contact Advantech’s applications engineer to obtain an up-to-date patch code file. This will ensure that your CPU’s system status is valid. After ensuring that you have a number assigned to the patch code, pressing <Del> to allow you to enter the setup

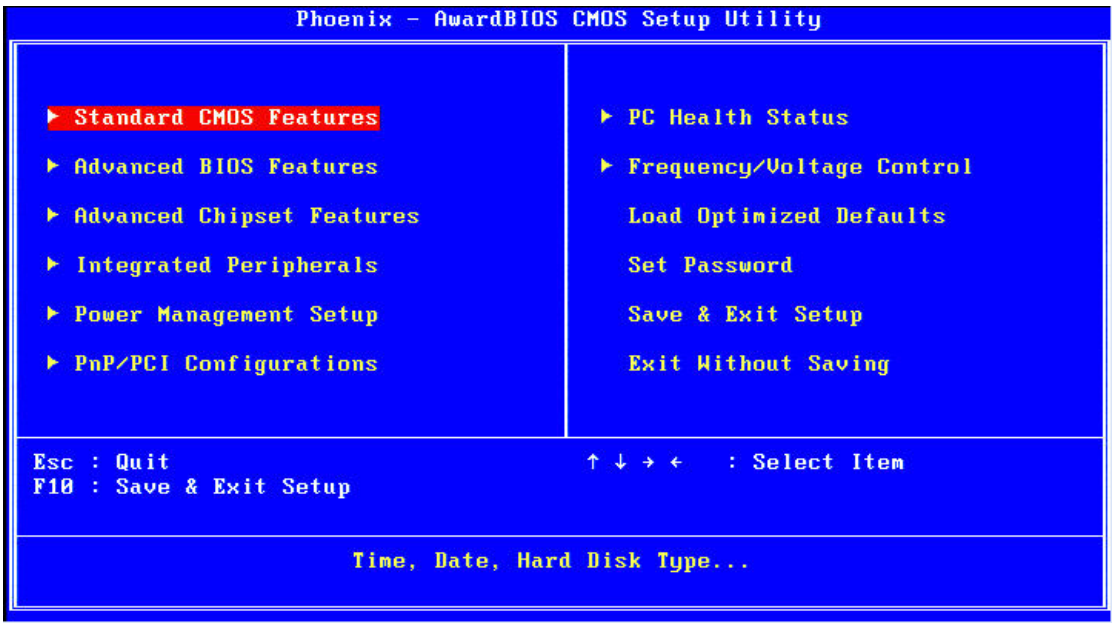


Figure 4.1: Award BIOS Setup initial screen

4.3 Standard CMOS Setup

Choose the “Standard CMOS Features” option from the “Initial Setup Screen” menu, and the screen below will be displayed. This menu allows users to configure system components such as date, time, hard disk drive, floppy drive, display, and memory.

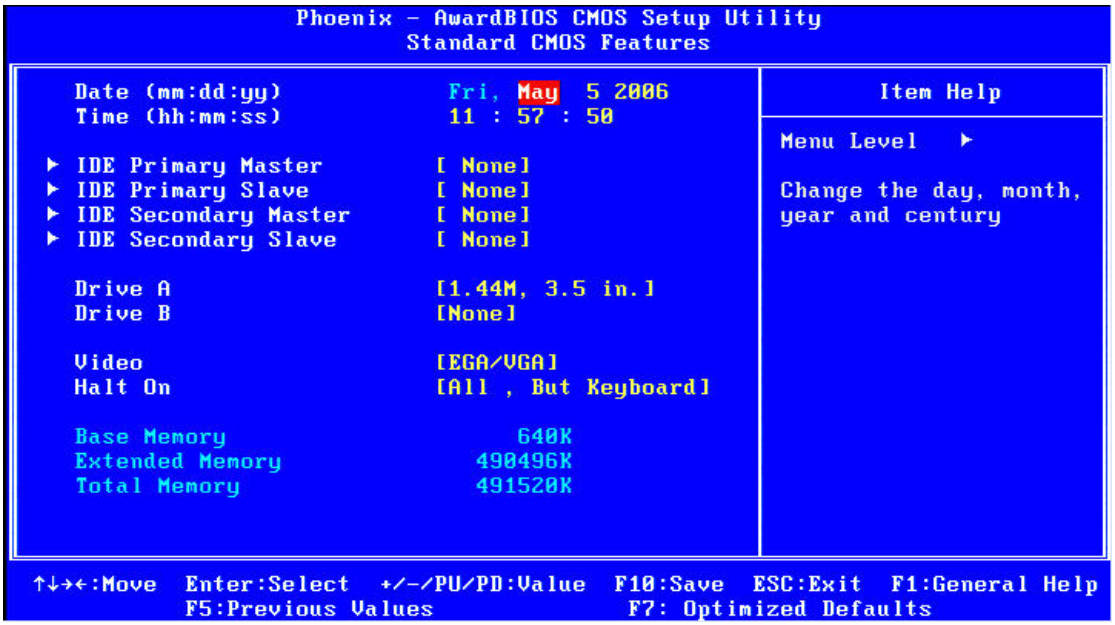


Figure 4.2: Standard CMOS features screen

4.4 Advanced BIOS Features

The “Advanced BIOS Features” screen appears when choosing the “Advanced BIOS Features” item from the “Initial Setup Screen” menu. It allows the user to configure the DVS-350 according to his particular requirements. Below are some major items that are provided in the Advanced BIOS Features screen. A quick booting function is provided for your convenience. Simply enable the Quick Booting item to save yourself valuable time

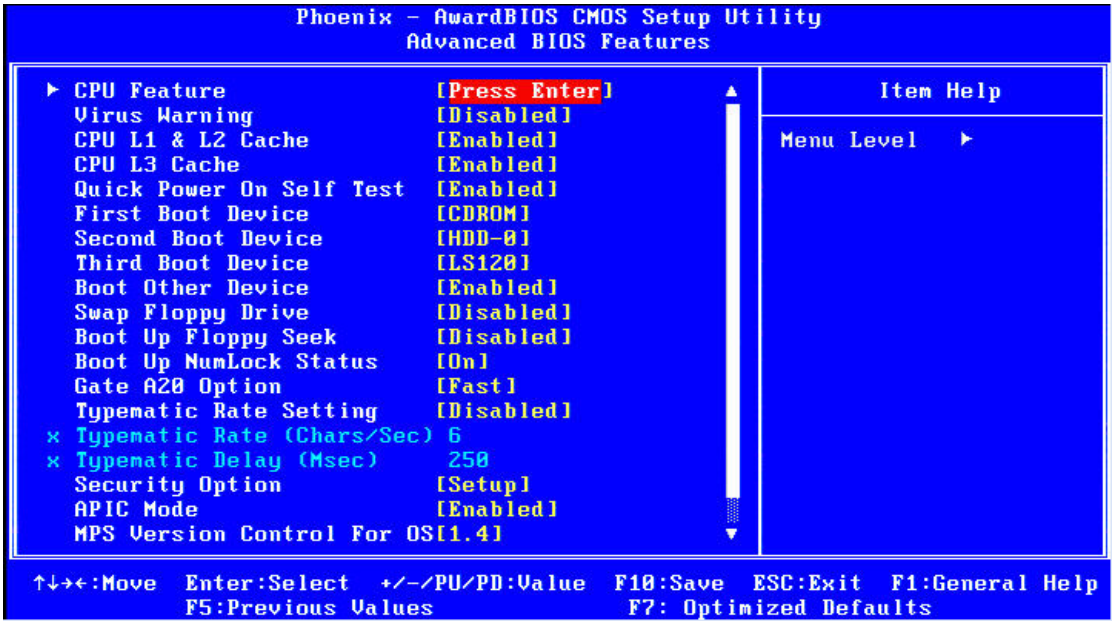


Figure 4.3: Advanced BIOS features screen

4.4.1 CPU Feature

Press Enter to configure the settings relevant to CPU Feature.

4.4.2 Virus Warning

If enabled, a warning message and alarm beep activates if someone attempts to write here. The commands are “Enabled” or “Disabled.”

4.4.3 CPU L1 & L2 Cache

Enabling this feature speeds up memory access. The commands are “Enabled” or “Disabled.”

4.4.4 Quick Power On Self Test

This option speeds up the Power-On Self Test (POST) conducted as soon as the computer is turned on. When enabled, BIOS shortens or skips some of the items during the test. When disabled, the computer conducts normal POST procedures.

4.4.5 First/Second/Third/ Boot Other Device

The BIOS tries to load the OS with the devices in the sequence selected. Choices are: Floppy, LS/ZIP, HDD, SCSI, CDROM, LAN, Disabled.

4.4.6 Swap Floppy Drive

Logical name assignments of floppy drives can be swapped if there is more than one floppy drive. The commands are “Enabled” or “Disabled.”

4.4.7 Boot UP Floppy Seek

Selection of the command “Disabled” will speed the boot up. Selection of “Enabled” searches disk drives during boot up.

4.4.8 Boot Up NumLock Status

This feature selects the “power on” state for NumLock. The commands are “Enabled” or “Disabled.”

4.4.9 Gate A20 Option

Normal: A pin in keyboard controller controls GateA20

Fast (Default): Chipset controls GateA20

The typematic rate is the rate key strokes repeat as determined by the keyboard controller. The commands are “Enabled” or “Disabled.” Enabling allows the typematic rate and delay to be selected.

4.4.10 Typematic Rate (Chars/Sec)

BIOS accepts the following input values (characters/second) for typematic rate: 6, 8, 10, 12, 15, 20, 24, 30.

4.4.11 Typematic Delay (msec)

Typematic delay is the time interval between the appearance of two consecutive characters, when holding down a key. The input values for this category are: 250, 500, 750, 1000 (msec).

4.4.12 Security Option

This field allows you to limit access to the System and Setup. The default value is Setup. When you select System, the system prompts for the User Password every time you boot up. When you select Setup, the system always boots up and prompts for the Supervisor Password only when the Setup utility is called up.

4.4.13 APIC Mode

APIC stands for Advanced Programmable Interrupt Controller. The default setting is Enabled.

4.4.14 MPS Version Control For OS

This option specifies the MPS (Multiprocessor Specification) version for your operating system. MPS version 1.4 added extended configuration tables to improve support for multiple PCI bus configurations and improve future expandability. The default setting is 1.4.

4.4.15 OS Select for DRAM > 64MB

This option allows the system to access greater than 64MB of DRAM memory when used with OS/2 that depends on certain BIOS calls to access memory. The default setting is Non-OS/2.

4.4.16 Report No FDD For WIN 95

If you are using Windows 95/98 without a floppy disk drive, select Enabled to release IRQ6. This is required to pass Windows 95/98's SCT test. You should also disable the Onboard FDC Controller in the Integrated Peripherals screen when there's no floppy drive in the system. If you set this feature to Disabled, the BIOS will not report the missing floppy drive to Win95/98.

4.4.17 Small Logo (EPA) Show

The EPA logo appears at the right side of the monitor screen when the system is boot up. The default setting is Enabled.

4.5 Advanced Chipset Features

The “Advanced Chipset Features” screen appears when choosing the “Advanced Chipset Features” item from the “Initial Setup Screen” menu. It allows the user to configure the system chipset according to his particular requirements. Below are some major items that are provided in the Advanced Chipset Features screen.

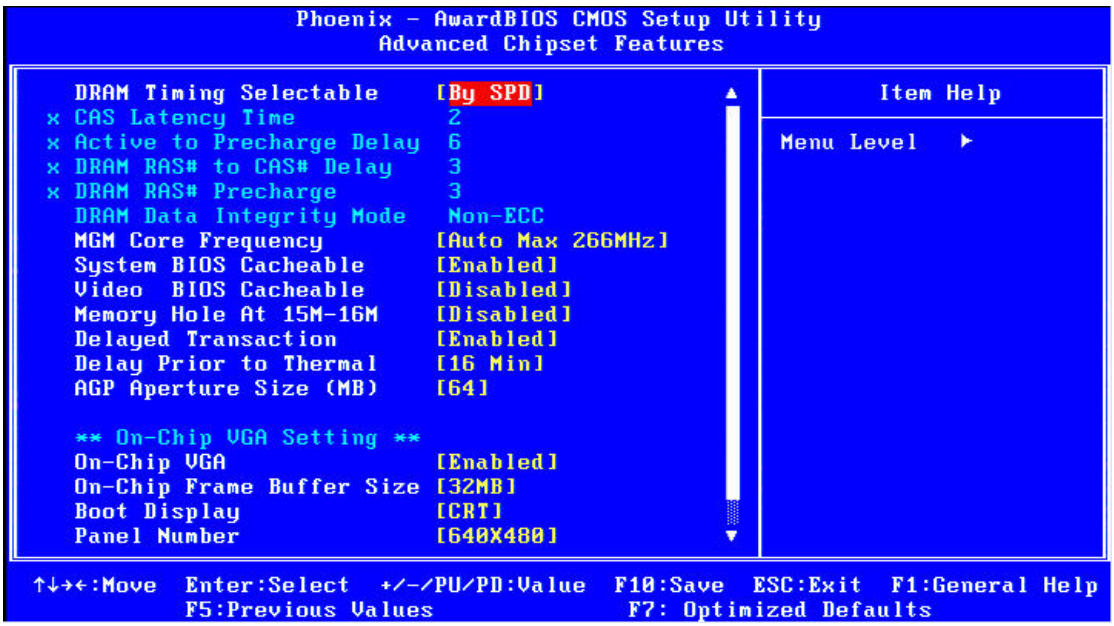


Figure 4.4: Advanced Chipset features screen

4.5.1 DRAM Timing Selectable

This option refers to the method by which the DRAM timing is selected. The default is By SPD.

4.5.2 CAS Latency Time

You can configure CAS latency time in HCLKs as 2 or 2.5 or 3. The system board designer should set the values in this field, depending on the DRAM installed. Do not change the values in this field unless you change specifications of the installed DRAM or the installed CPU.

4.5.3 Active to Precharge Delay

The default setting for the Active to Precharge Delay is 7.

4.5.4 DRAM RAS# to CAS# Delay

This option allows you to insert a delay between the RAS (Row Address Strobe) and CAS (Column Address Strobe) signals. This delay occurs when the SDRAM is written to, read from or refreshed. Reducing the delay improves the performance of the SDRAM.

4.5.5 DRAM RAS# Precharge

This option sets the number of cycles required for the RAS to accumulate its charge before the SDRAM refreshes. The default setting for the Active to Precharge Delay is 3.

4.5.6 DRAM Data Integrity Mode

Select ECC if your memory module supports it. The memory controller will detect and correct single-bit soft memory errors. The memory controller will also be able to detect double-bit errors though it will not be able to correct them. This provides increased data integrity and system stability.

4.5.7 MGM Core Frequency

This field sets the frequency of the DRAM memory installed. The default setting is Auto Max 266MHz.

4.5.8 System BIOS Cacheable

The setting of Enabled allows caching of the system BIOS ROM at F0000h-FFFFFh, resulting in better system performance. However, if any program writes to this memory area, a system error may result.

4.5.9 Video BIOS Cacheable

The Setting Enabled allows caching of the video BIOS ROM at C0000h-C7FFFh, resulting in better video performance. However, if any program writes to this memory area, a system error may result.

4.5.10 Memory Hole At 15M-16M

In order to improve performance, certain space in memory can be reserved for ISA cards. This memory must be mapped into the memory space below 16 MB. The choices are Enabled and Disabled.

4.5.11 Delayed Transaction

The chipset has an embedded 32-bit posted write buffer to support delay transactions cycles. Select Enabled to support compliance with PCI specification version 2.1.

4.5.12 Delay Prior to Thermal

This field activates the CPU thermal function after the systems boots for the set number of minutes. The options are 16Min and 64Min.

4.5.13 AGP Aperture Size (MB)

The field sets aperture size of the graphics. The aperture is a portion of the PCI memory address range dedicated for graphics memory address space. Host cycles that hit the aperture range are forwarded to the AGP without any translation. The default setting is 64M.

4.5.14 On-Chip VGA

The default setting is Enabled.

4.5.15 On-Chip Frame Buffer Size

The default setting is 32MB. The options available include 1MB, 4MB, 8MB and 16MB.

4.5.16 Boot Display

The default setting is CRT+LVDS. The options available include CRT, LVDS, DVI and TV.

4.5.17 Panel Number

These fields allow you to select the LCD Panel type. The setting values for these ports are:

640 x 480 18bit SC

800 x 600 18bit SC

1024 x 768 18bit SC

1280 x 1024 24bit SC

1400 x 1050 18bit SC

1024 x 768 24bit SC

1600 x 1200 24bit SC

4.5.18 TV Standard

The default setting is PAL. The options available include PAL and NTSC.

4.6 Integrated Peripherals

This section sets configurations for your hard disk and other integrated peripherals. The first screen shows three main items for user to select. Once an item selected, a submenu appears. Details as follows.

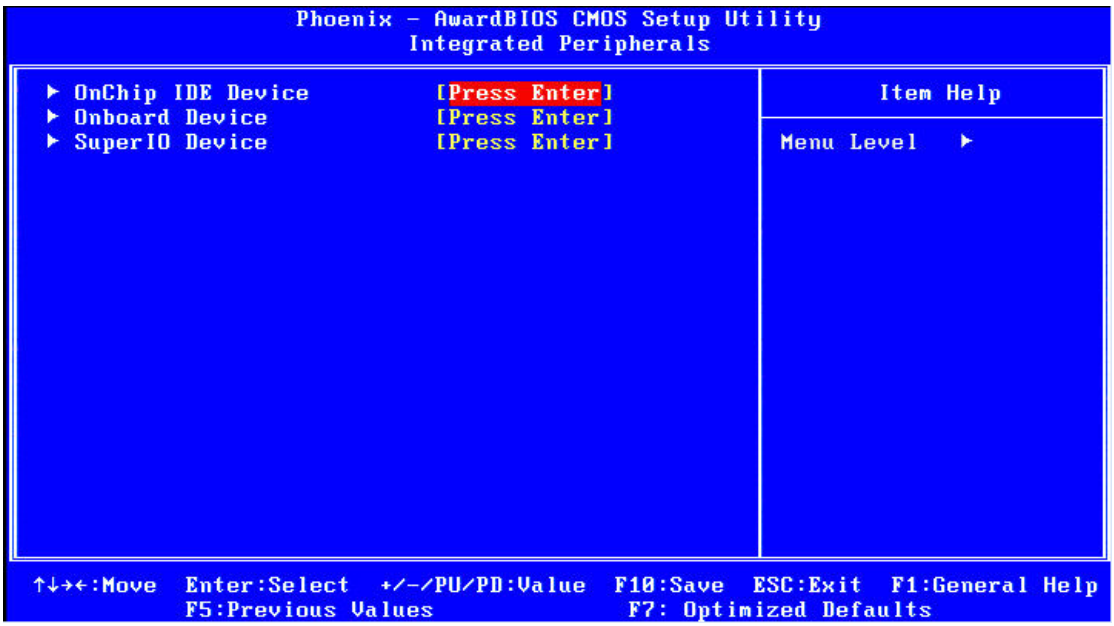


Figure 4.5.1: Integrated Peripherals

4.6.1 On-Chip Primary PCI IDE

Some UDMA cables use a hole in the ribbon cable as a cable detect mechanism to determine if a UDMA IDE or standard IDE cable is installed. The default setting is “Enabled”.

4.6.2 IDE Master/Slave PIO/UDMA Mode,

IDE Primary (Secondary) Master/Slave PIO/UDMA Mode (Auto) Each channel (Primary and Secondary) has both a master and a slave, making four IDE devices possible. Because each IDE device may have a different Mode timing (0, 1, 2, 3, 4), it is necessary for these to be independent. The default setting “Auto” will allow auto-detection to ensure optimal performance.

4.6.3 On-Chip Secondary PCI IDE

If you enable IDE HDD Block Mode, the enhanced IDE driver will be enabled. Leave IDE HDD Block Mode on the default setting.

4.6.4 IDE HDD Block Mode

You can enable the Primary IDE channel and/or the Secondary IDE channel. Any channel not enabled is disabled. This field is for systems with only SCSI drives.

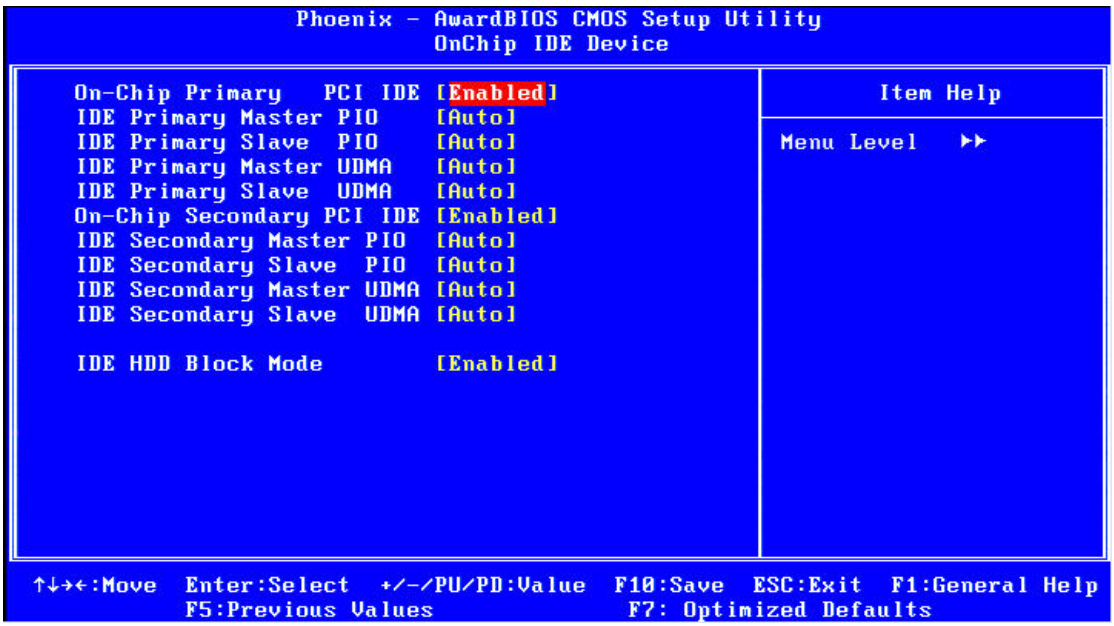


Figure 4.5.2: OnChip IDE Device

4.6.5 USB/USB 2.0 Controller

Select Enabled if your system contains a Universal Serial Bus (USB) controller and you have USB peripherals. The choices: Enabled, Disabled.

4.6.6 USB Keyboard/Mouse Support

Select Enabled if user plan to use an USB keyboard. The choice: Enabled, Disable.

4.6.7 AC97 Audio/Modem

Select Disable if you do not want to use AC-97 audio/Modem. Option is Auto, Disable.

4.6.8 Init Display First

This item allows you to choose which one to activate first, PCI Slot or onchip VGA. The choices: PCI Slot, Onboard/AGP.

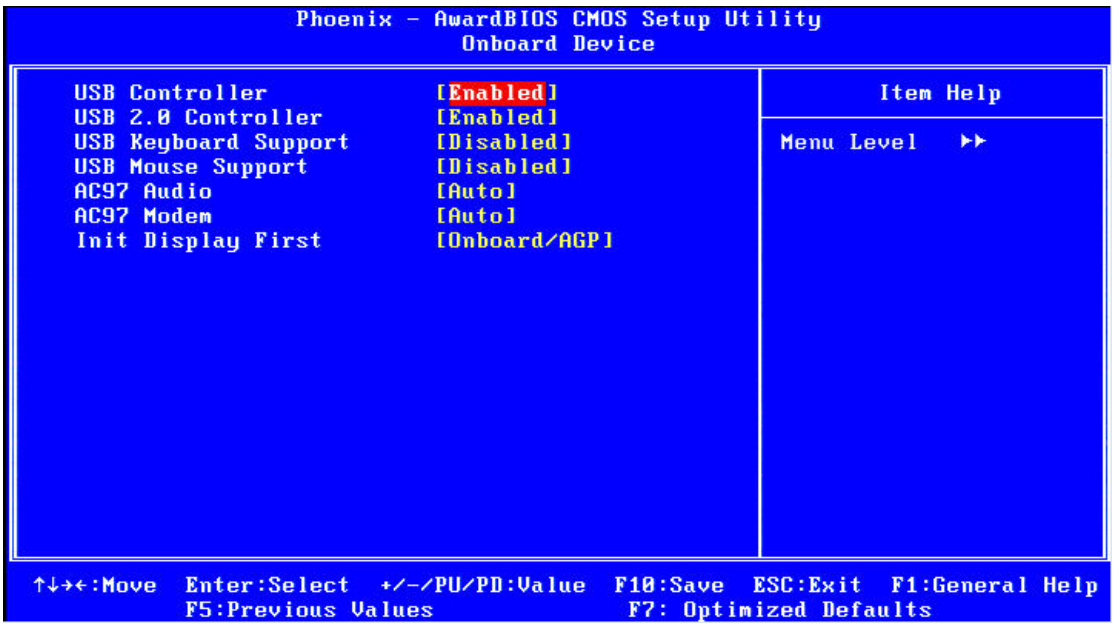


Figure 4.5.3: Onboard Device

4.6.9 Onboard LAN Control

Option is Enable and Disable. Select Disable if user does not want to use onboard LAN controller

4.6.10 Onboard FDC Controller

When enabled, this field allows you to connect your floppy disk drives to the onboard floppy disk drive connector instead of a separate controller card. If you want to use a different controller card to connect the floppy disk drives, set this field to Disabled.

4.6.11 Onboard Serial Port

For settings reference the Appendix for the serial resource allocation, and Disabled for the on-board serial connector.

4.6.12 UART Mode Select

This item allows you to select UART mode. The choices: IrDA, ASKIR, Normal.

4.6.13 RxD, TxD Active

This item allows you to determine the active of RxD, TxD. The Choices: “Hi, Hi,” “Lo, Lo,” “Lo, Hi,” “Hi, Lo.”

4.6.14 IR Transmission Delay

This item allows you to enable/disable IR transmission delay. The choices: Enabled, Disabled.

4.6.15 UR2 Duplex Mode

This item allows you to select the IR half/full duplex function. The choices: Half, Full.

4.6.16 Onboard Parallel Port

This field sets the address of the on-board parallel port connector. You can select either 3BCH/IRQ7, 378H/IRQ7, 278H/IRQ5 or Disabled. If you install an I/O card with a parallel port, make sure there is no conflict in the address assignments. The CPU card can support up to three parallel ports, as long as there are no conflicts for each port.

4.6.17 Parallel Port Mode

This field allows you to set the operation mode of the parallel port. The setting “Normal” allows normal speed operation, but in one direction only. “EPP” allows bi-directional parallel port operation at maximum speed. “ECP” allows the parallel port to operate in bi-directional mode and at a speed faster than the maximum data transfer rate. “ECP + EPP” allows normal speed operation in a two-way mode.

4.6.18 EPP Mode Select

This field allows you to select EPP port type 1.7 or 1.9. The choices: EPP1.7, 1.9.

4.6.19 ECP Mode Use DMA

This selection is available only if you select “ECP” or “ECP + EPP” in the Parallel Port Mode field. In ECP Mode Use DMA, you can select DMA channel 1, DMA channel 3, or Disable. Leave this field on the default setting.

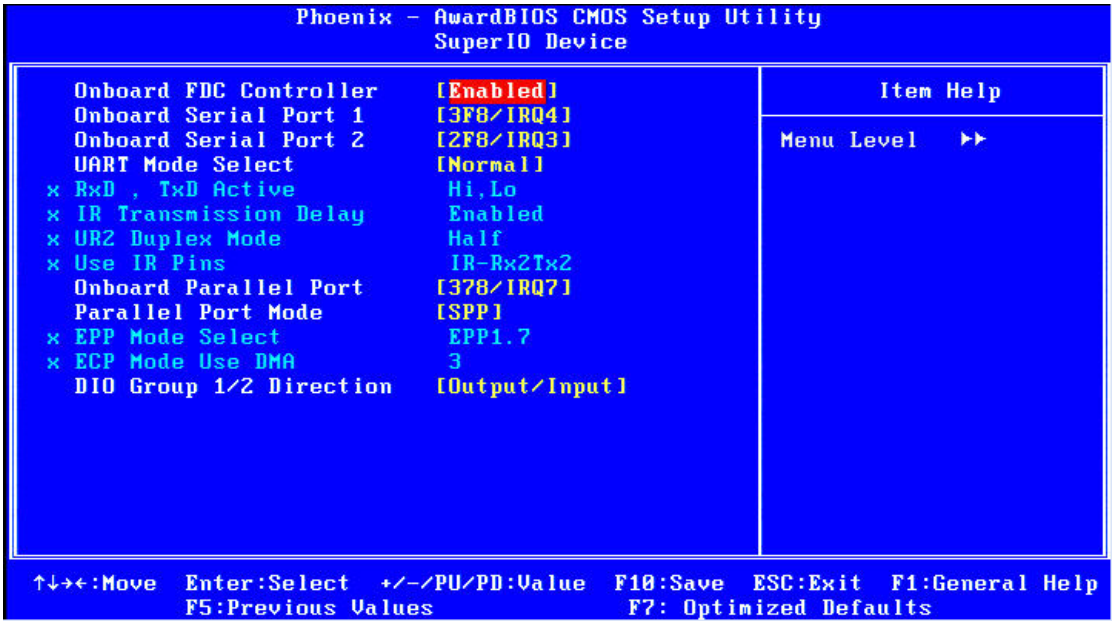


Figure 4.5.4: SuperIO Device

4.7 Power Management Setup

The power management setup controls the CPU card's "green" features to save power. The following screen shows the manufacturer's defaults:

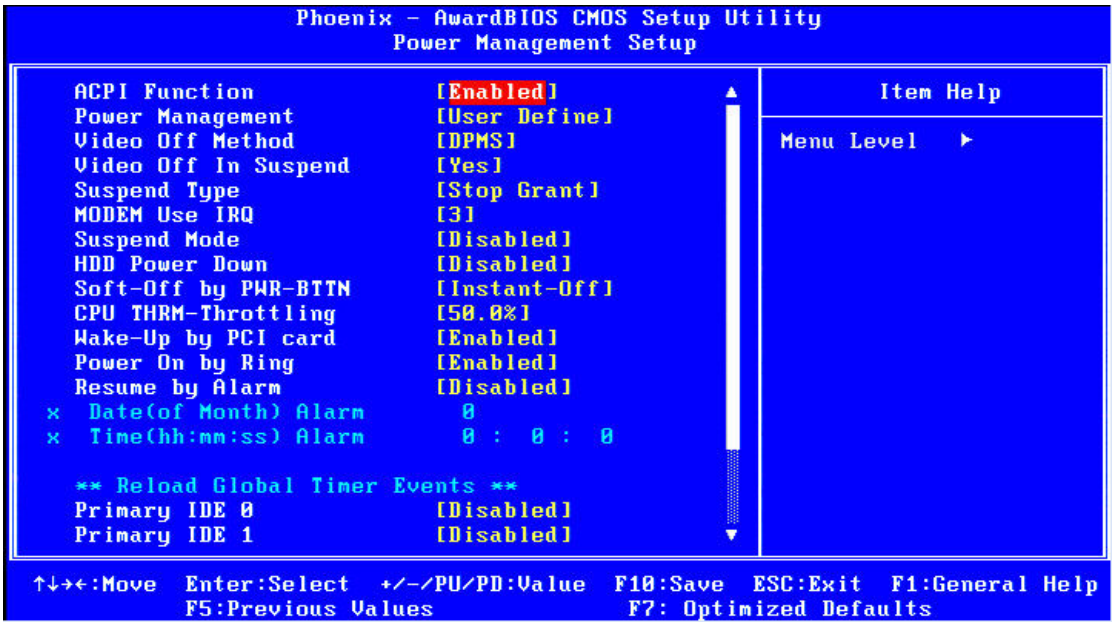


Figure 4.6: Power management setup screen

4.7.1 ACPI function

The choice: Enabled, Disabled.

4.7.2 Power Management

This category allows you to select the type (or degree) of power saving and is directly related to the following modes:

1. HDD Power Down
2. Suspend Mode

There are four selections for Power Management, three of which have fixed mode settings

Min. Power Saving	Minimum power management., Suspend Mode = 1 hr., and HDD Power Down = 15 min.
Max. Power Saving	Maximum power management., Suspend Mode = 1 min., and HDD Power Down = 1 min.
User Defined (Default)	Allows you to set each mode individually. When not disabled, each of the ranges are from 1 min. to 1 hr. except for HDD Power Down which ranges from 1 min. to 15 min. and disable.

4.7.3 Video Off In Suspend

When system is in suspend, video will turn off.

4.7.4 Modem Use IRQ

This determines the IRQ in which the MODEM can use.The choices: 3, 4, 5, 7, 9, 10, 11, NA.

4.7.5 HDD Power Down

You can choose to turn the HDD off after one of the time intervals listed, or when the system is in “suspend” mode. If the HDD is in a power saving mode, any access to it will wake it up.

4.7.6 Soft-Off by PWR-BTTN

If you choose “Instant-Off”, then pushing the ATX soft power switch button once will switch the system to “system off” power mode. You can choose “Delay 4 sec.” If you do, then pushing the button for more than 4 seconds will turn off the system, whereas pushing the button momentarily (for less than 4 seconds) will switch the system to “suspend” mode.

4.7.7 CPU THRM-Throttling

This field allows you to select the CPU THRM-Throttling rate. The choices: 12.5%, 24.0%, 37.5%, 50.0%, 62.5%, 74.0%, 87.5%.

4.7.8 PowerOn By LAN

This item allows you to wake up the system via LAN from the remote host. The choices: Enabled, Disabled.

4.7.9 PowerOn By Modem

When Enabled an input signal on the serial Ring Indicator (RI) line (in other words, an incoming call on the modem) awakens the system from a soft off state. The choices: Enabled, Disabled.

4.7.10 PowerOn By Alarm

When Enabled, your can set the date and time at which the RTC (real time clock) alarm awakens the system from Suspend mode. The choices: Enabled, Disabled.

4.7.11 Primary IDE 0 (1) and Secondary IDE 0 (1)

When Enabled, the system will resume from suspend mode if Primary IDE 0 (1) or Secondary IDE 0 (1) is active. The choice: Enabled, Disabled.

4.7.12 FDD, COM, LPT PORT

When Enabled, the system will resume from suspend mode if FDD, COM port, or LPT port is active. The choice: Enabled, Disabled.

4.7.13 PCI PIRQ [A-D]#

When Enabled, the system will resume from suspend mode if interrupt occurs. The choice: Enabled, Disabled.

4.8 PnP/PCI Configurations

4.8.1 PnP OS Installed

Select “Yes” if you are using a plug and play capable operating system.

Select No if you need the BIOS to configure non-boot device

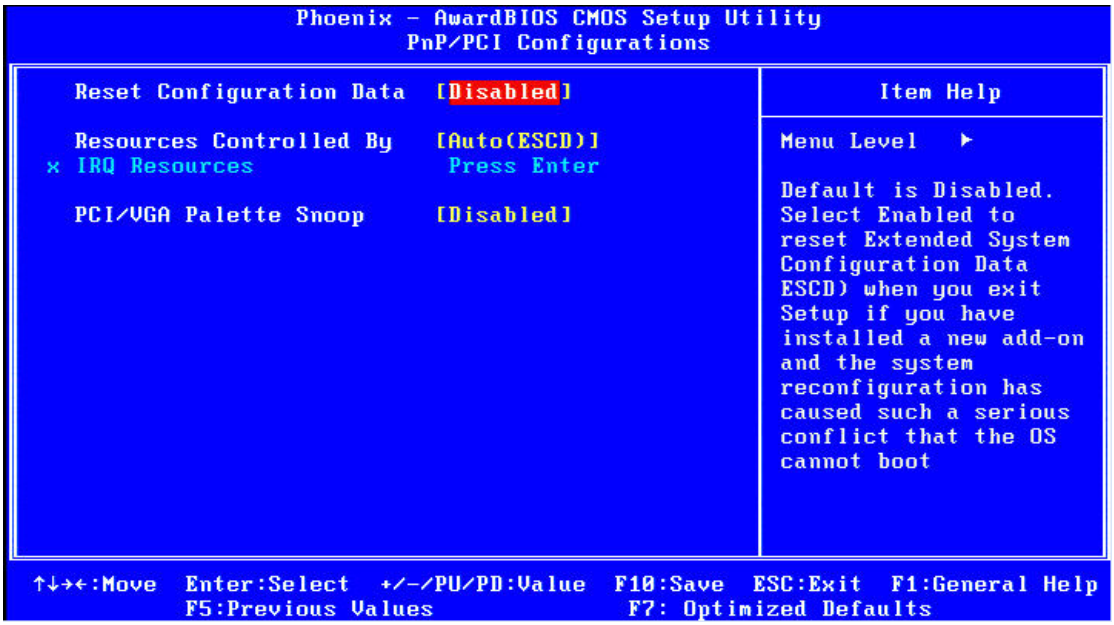


Figure 4.7: PnP/PCI configurations screen

4.8.2 Reset Configuration Data

Default is Disable. Select Enable to reset Extended System Configuration Data (ESCD) if you have installed a new add-on and system configuration has caused such a conflict that OS cannot boot.

4.8.3 Resources controlled by:

The commands here are “Auto” or “Manual.” Choosing “manual” requires you to choose resources from each following sub-menu. “Auto” automatically configures all of the boot and Plug and Play devices but you must be using Windows 95 or above.

4.8.4 PCI/VGA Palette Snoop

This is left at “Disabled.”

4.9 Password Setting

To change the password:

1. Choose the “Set Password” option from the “Initial Setup Screen” menu and press <Enter>. The screen will display the following message:

Please Enter Your Password

Press <Enter>.

2. If the CMOS is good or if this option has been used to change the default password, the user is asked for the password stored in the CMOS. The screen will display the following message:

Please Confirm Your Password

Enter the current password and press <Enter>.

3. After pressing <Enter> (ROM password) or the current password (user-defined), you can change the password stored in the CMOS. The password must be no longer than eight (8) characters.

Remember, to enable the password setting feature, you must first select either “Setup” or “System” from the “Advanced BIOS Features” menu.

4.10 Save & Exit Setup

If you select this and press <Enter>, the values entered in the setup utilities will be recorded in the CMOS memory of the chipset. The microprocessor will check this every time you turn your system on and compare this to what it finds as it checks the system. This record is required for the system to operate.

4.11 Exit Without Saving

Selecting this option and pressing <Enter> lets you exit the setup program without recording any new values or changing old ones.

CHAPTER 5

System Setup

This chapter details the needed driver installation for DVS-350

Sections include:

- Installation of chipset driver
- Installation of graphic driver
- Installation of LAN driver
- Installation of audio driver

Chapter 5 System Setup

5.1 Introduction

The system has an onboard Intel 852GM chipset for its graphic controller. It supports conventional analog CRT monitors and LCD displays with 64MB frame buffer shared with system memory. The VGA controller can drive CRT displays with resolutions up to 1600 x1200@85-Hz and 2048 x 536 @75Hz.

5.1.1 CMOS setting for panel type

The DVS-350 system BIOS and custom drivers are located in a Flash ROM device, designated U26 of system motherboard of DVS-350. A single Flash chip holds the system BIOS, VGA BIOS and network Boot ROM image. The display can be configured via CMOS settings. This method Choice of “Boot display” selection items of Advanced Chipset Features sections of Award BIOS Setup.

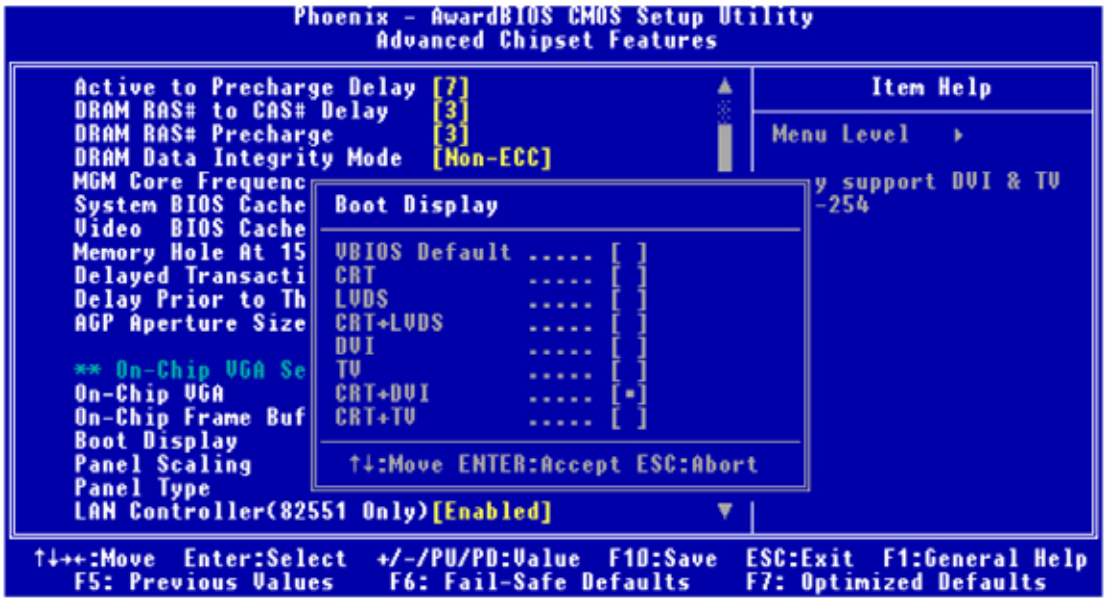


Figure 5.1: Advanced Chipset features screen

5.1.2 Display type

The DVS-350 can be set only in a VBIOS Default, CRT and TV mode. The system is initially set to “VBIOS Default”.

5.1.3 Disable of Dual Independent Display

The DVS-350 uses an Intel Intel 852GM controller that is capable of providing multiple views and simultaneous display with mixed video and graphics on a flat panel and CRT.

5.2 Installation of needed driver

5.2.1 Installation of a licensed Windows OS

Before installing the needed driver, you should complete the installation of a licensed Windows operating system –WinXP, WinXP embedded or Win2K first.

5.2.2 Installation of chipset driver.

The chipset driver is under the path:
DVS-350_CD:\01_DVS-350_Driver\Intel_855GME_852GM\01_Chipset
Please follow and pay attention to the installation instructions which appear on your screen, and complete the chipset driver installation.

5.2.3 Install the graphic driver

Find the graphic driver from CD at the directory of DVS-350_CD:\
01_DVS-350_Driver\Intel_855GME_852GM\02_Graphic

Please follow and pay attention to the installation instructions which appear on your screen, and complete the graphic driver installation.

5.2.4 Install the LAN driver

Find the LAN driver from CD at the directory of DVS-350_CD:\
01_DVS-350_Driver\Lan-Intal Pro1000MT\82551QM

Please follow and pay attention to the installation instructions which appear on your screen, and complete the LAN driver installation.

5.2.5 Install the audio driver

Find the audio driver from CD at the directory of DVS-350_CD:\
01_DVS-350_Driver\Audio Driver\Realtek_AC97_ALC202

Please follow and pay attention to the installation instructions which appear on your screen, and complete the audio driver installation.

5.2.6 Install the WLAN driver

Find the audio driver from CD at the directory of DVS-350_CD:\
01_DVS-350_Driver\RAlink Wireless Network Card Driver

Please follow and pay attention to the installation instructions which appear on your screen, and complete the audio driver installation.

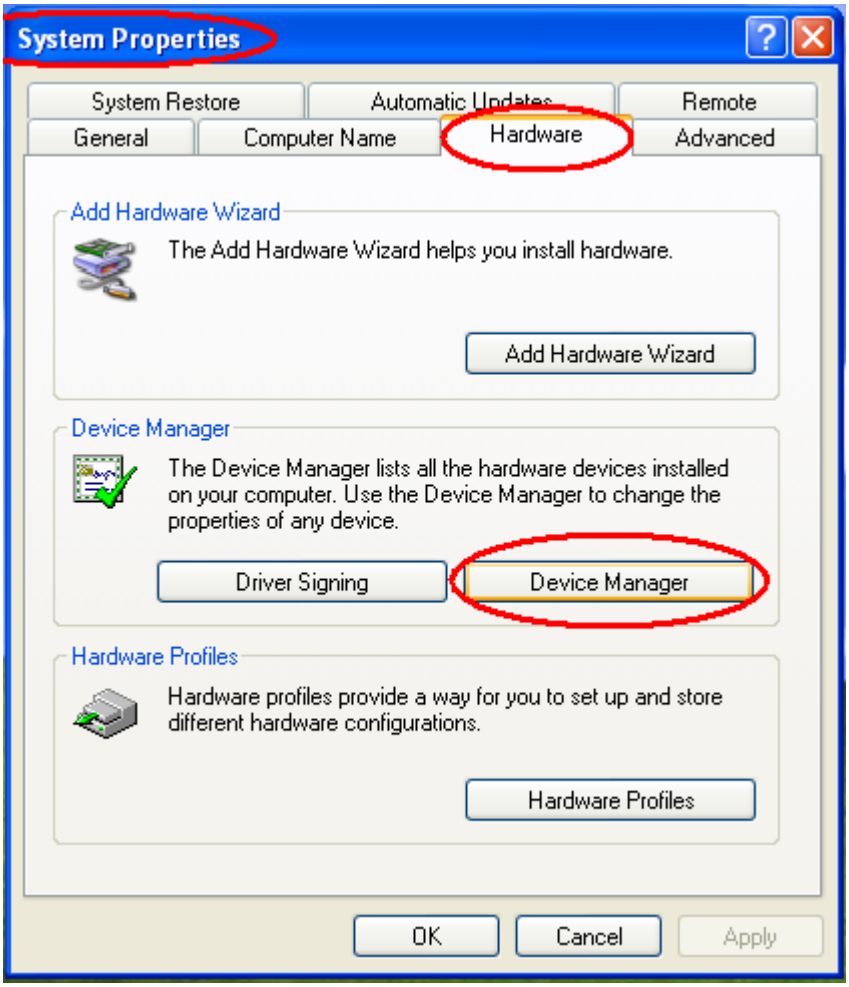
CHAPTER 6

Video capture installation

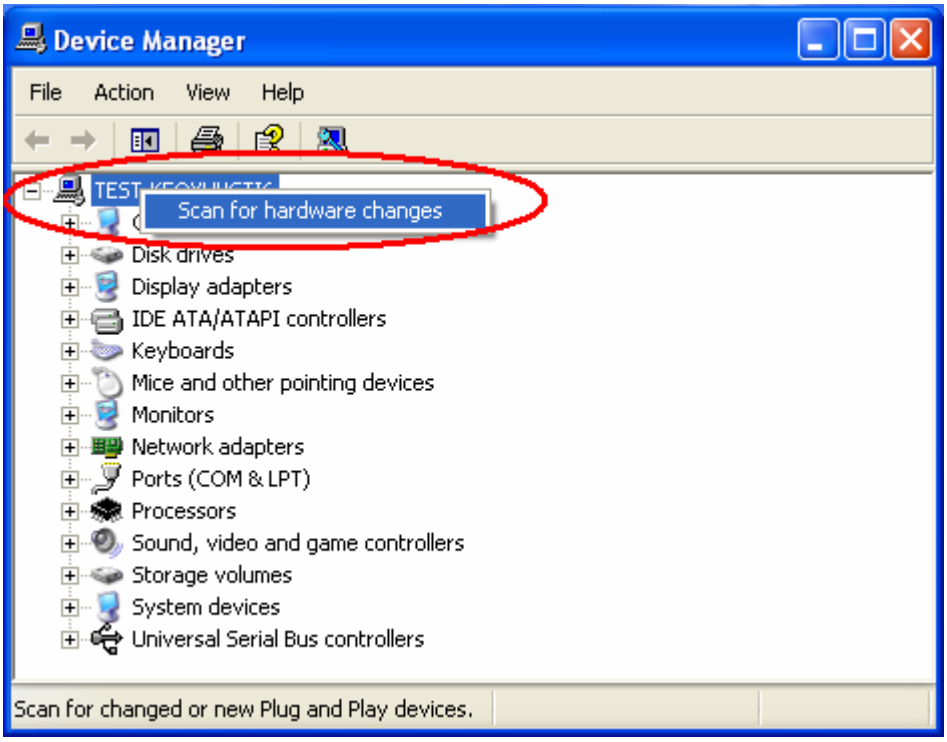
Chapter 6 Video capture installation

6.1 Driver installation of video capture chip

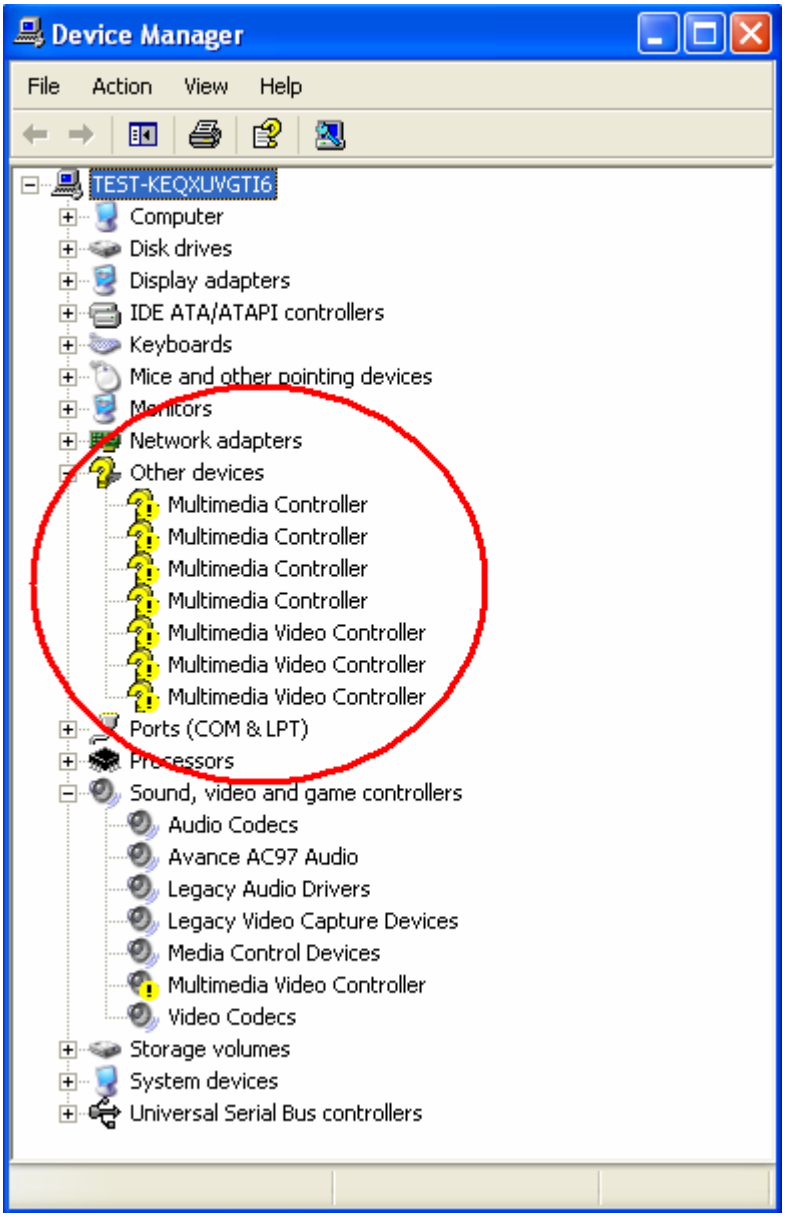
Step 1: Pop-up the “System Properties” window, choose the “Hardware” page, and press the “Device Manager” bottom.



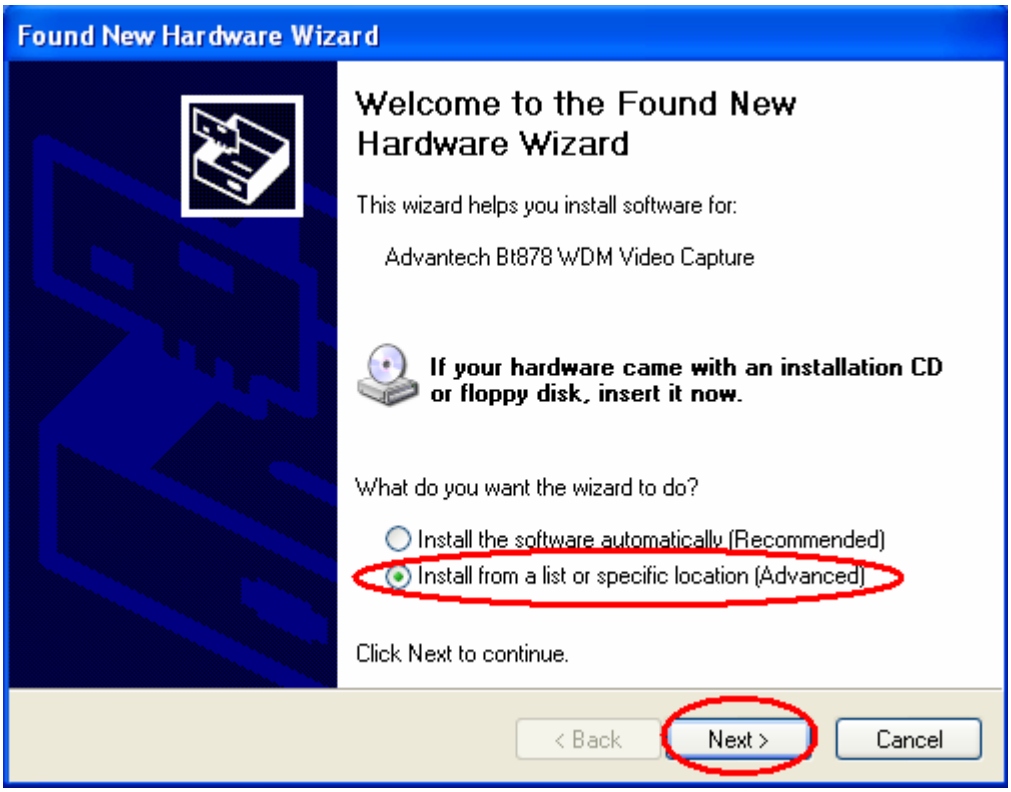
Step 2: Click the PC icon and press the left bottom of the mouse. Press the “Scan for hardware changes”.



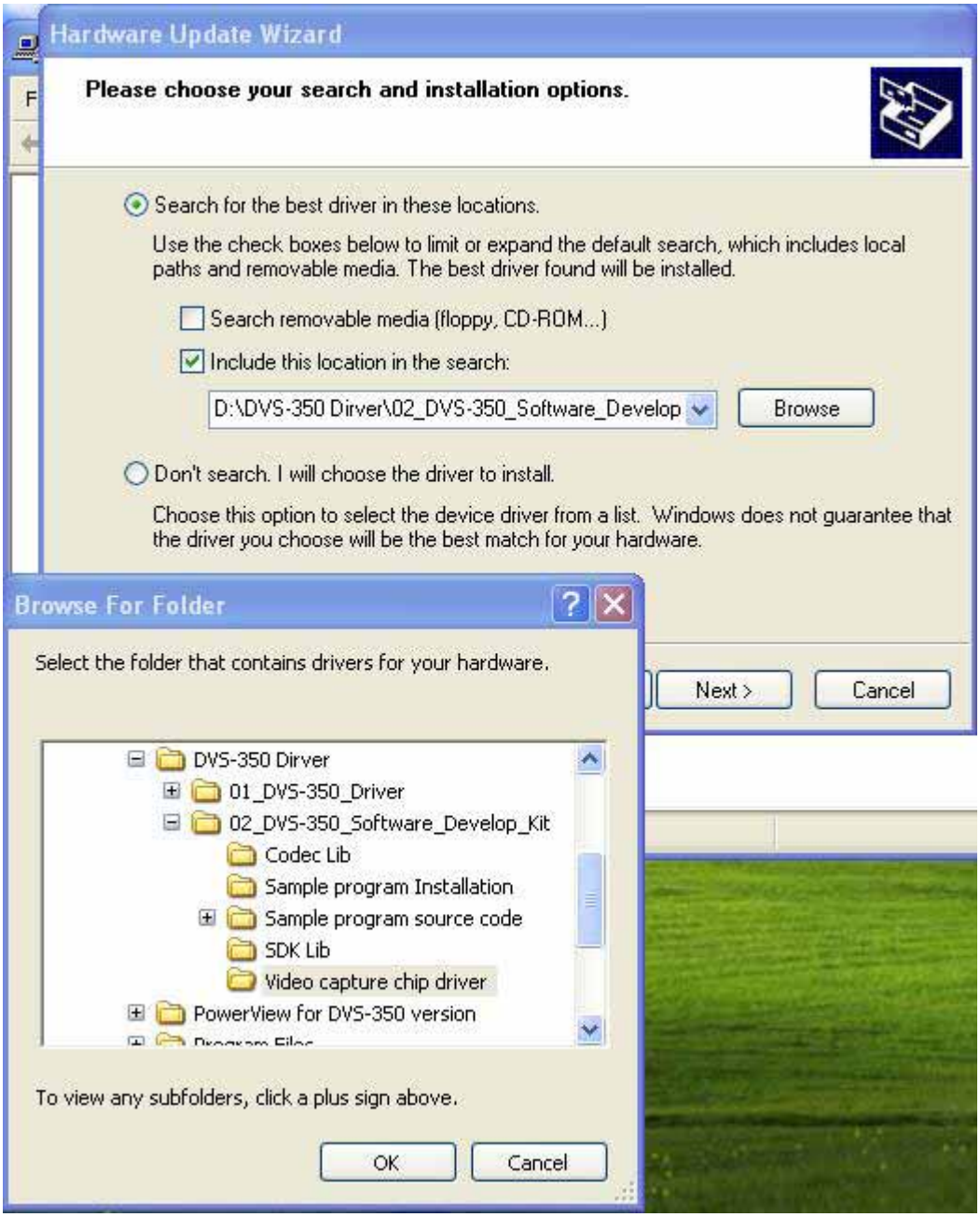
Step 3: The system will show the un-known devices like below window.



Step 4: Click the below icon to specify the driver location.

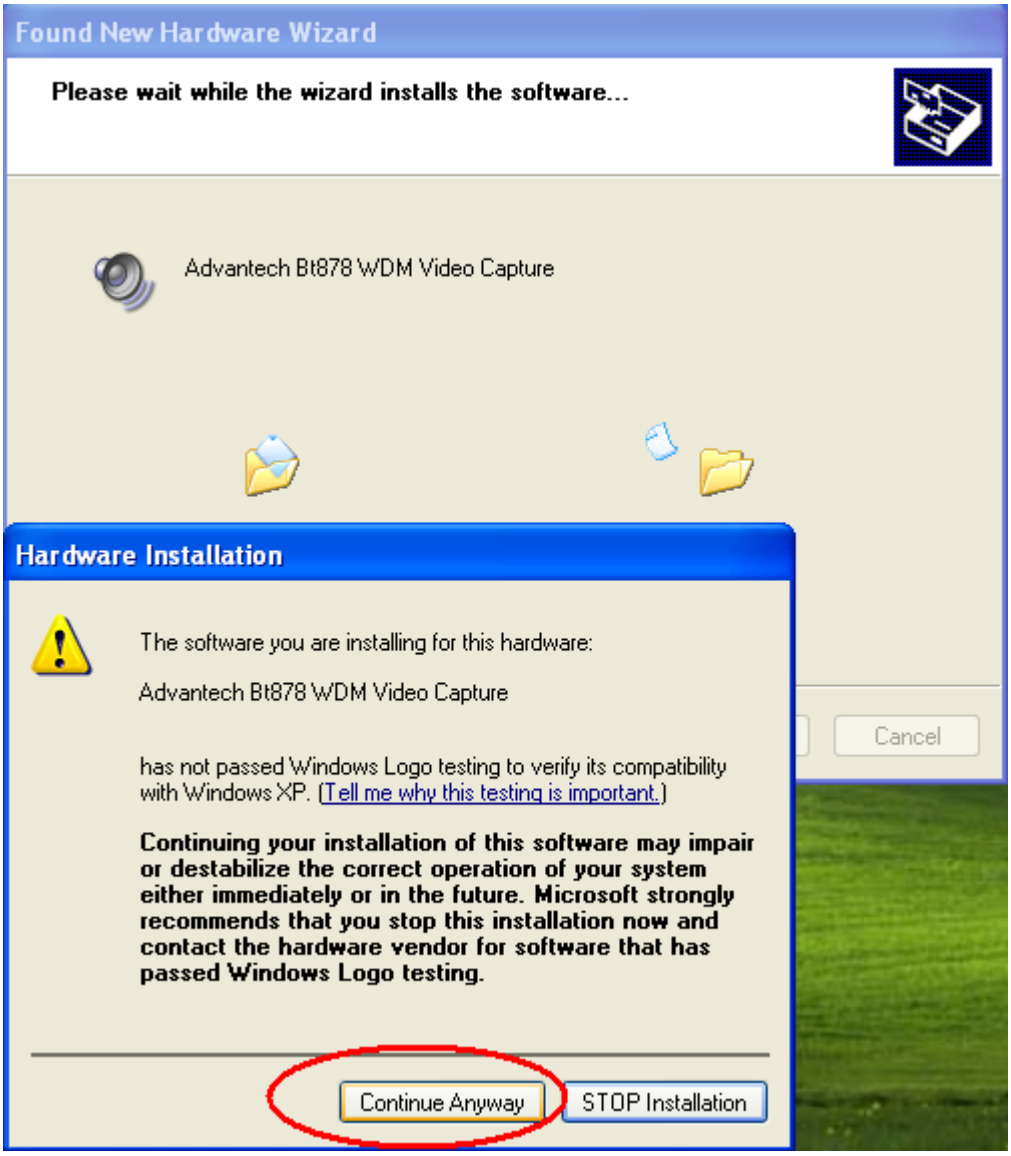


Step 5: Specify the driver under the DVS-350_CD\02_DVS-350_Software_Develop_Kit\Video capture chip driver

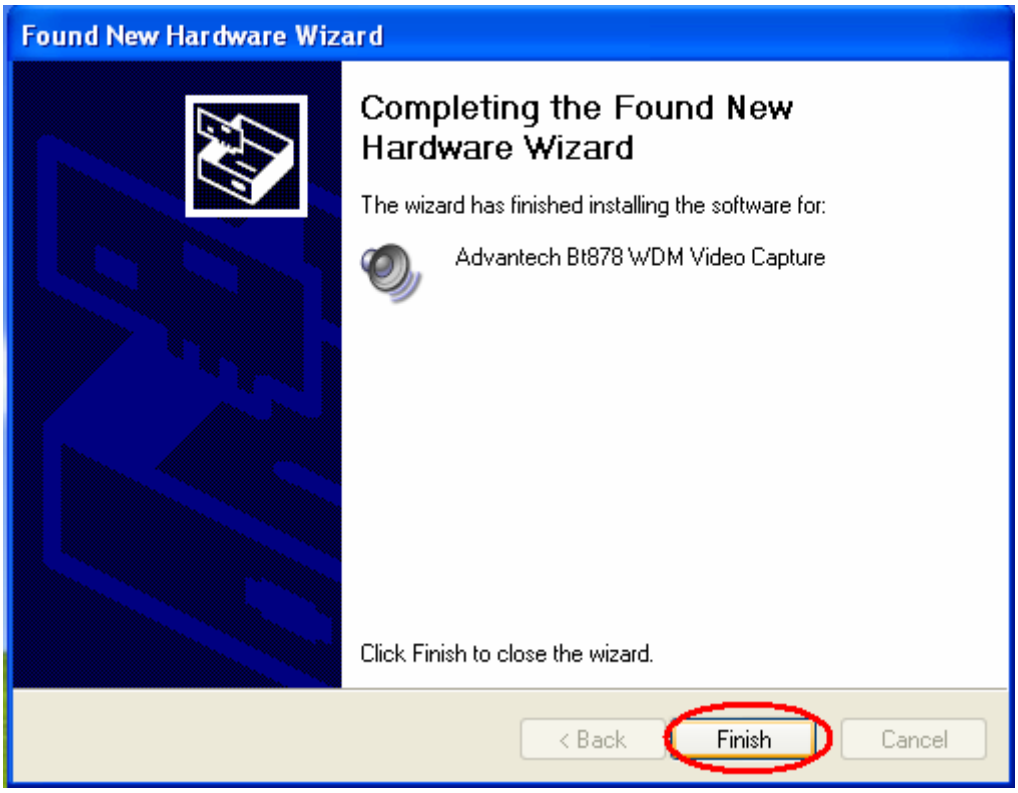


Step 6: Push the “Next” bottom to process the installation.

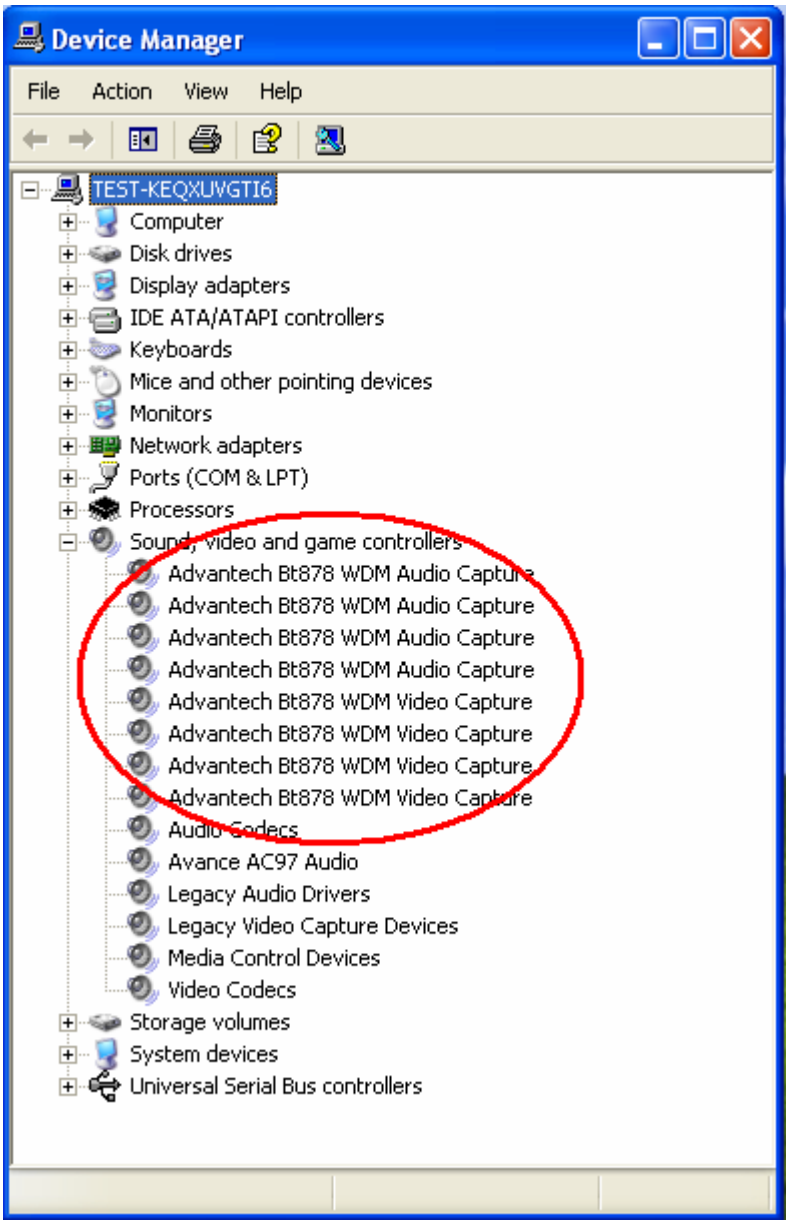
Step 7: Continuing the installation.



Step 8: Press the “Finish” bottom to finish the first circle installation. Then repeat the installation step 1~8 until all the un-known devices are all installed.

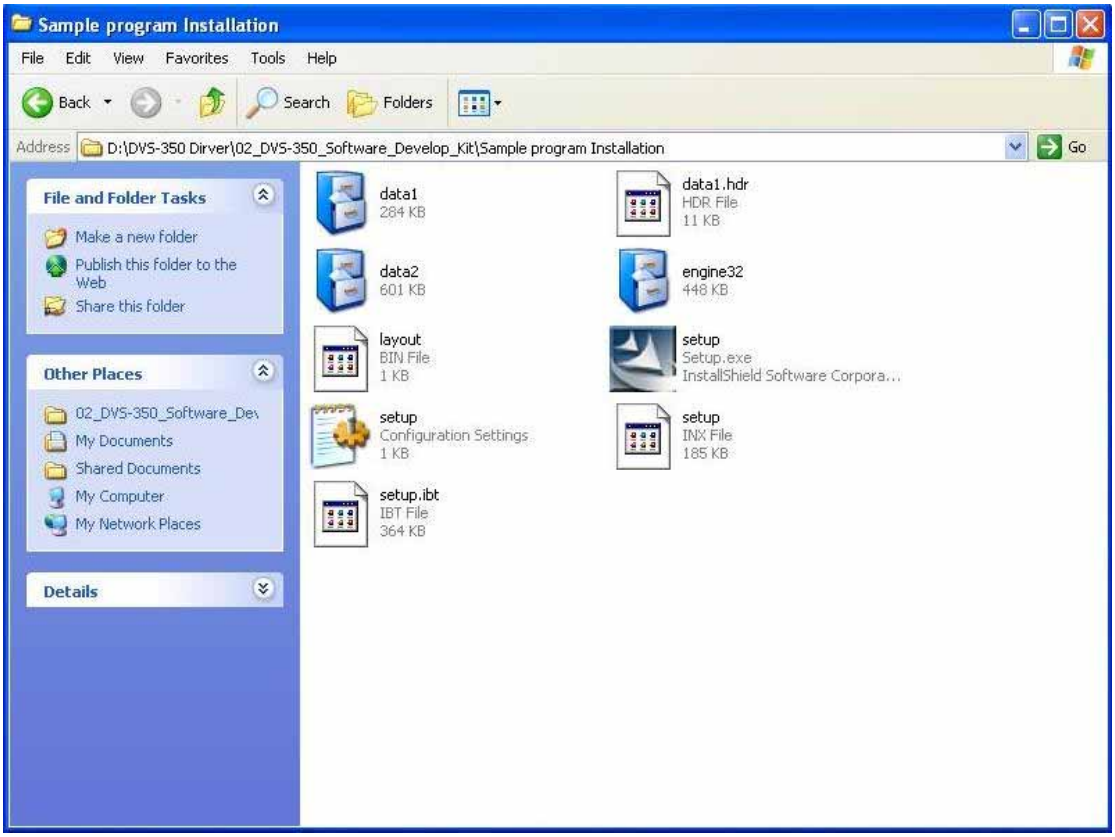


Step 9: From below window, we know there are 8 new items are installed.

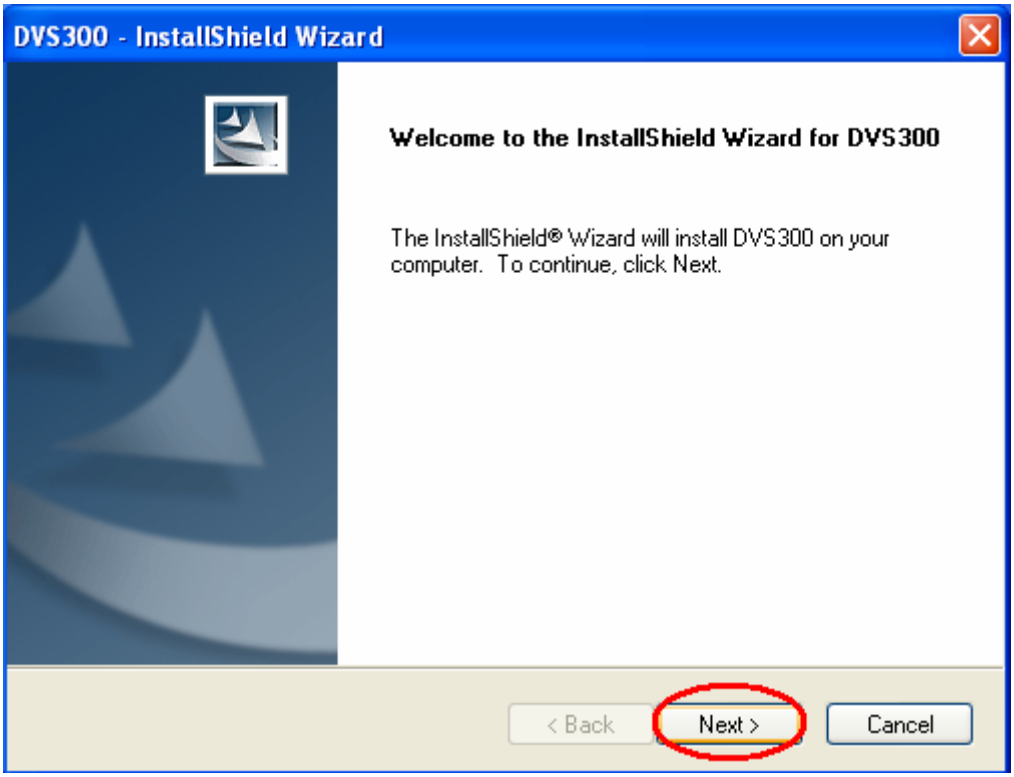


6.2 Installation of DVS-350 Demo Program

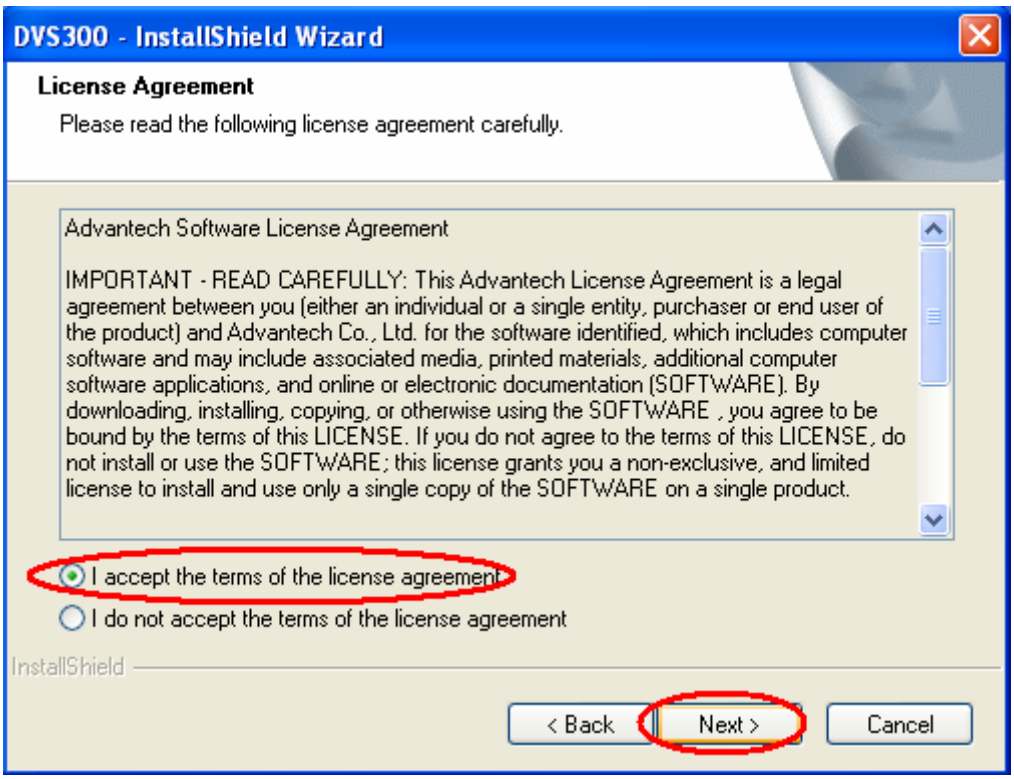
Step 1: Install the DVS-350 demo program. The executive file is in the path: DVS-350_CD\02_DVS-350_Software_Develop_Kit\Sample program Installation



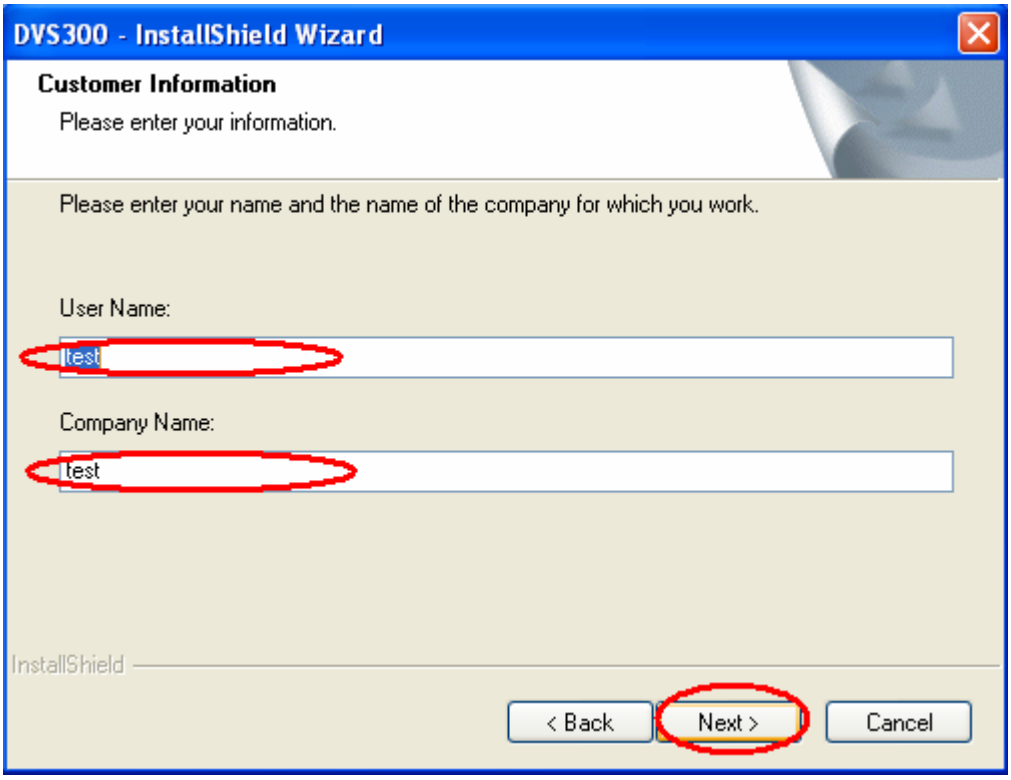
Step 2: Press the “Next” bottom to begin the installation.



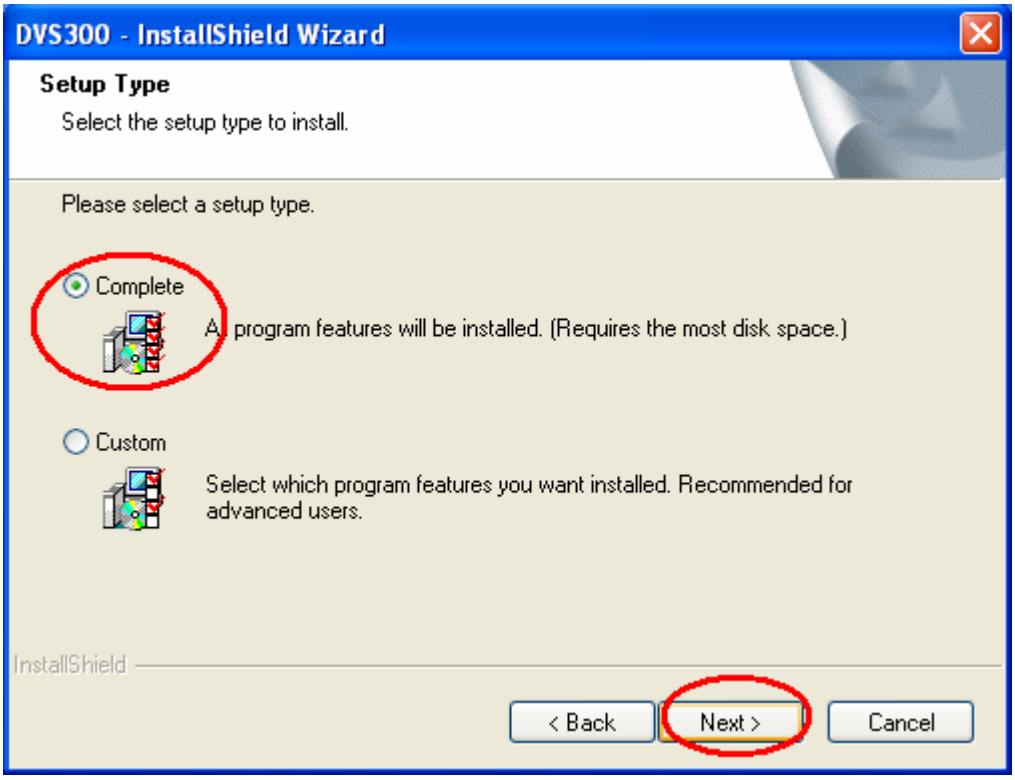
Step 3: Accept the license agreement and continue the installation.



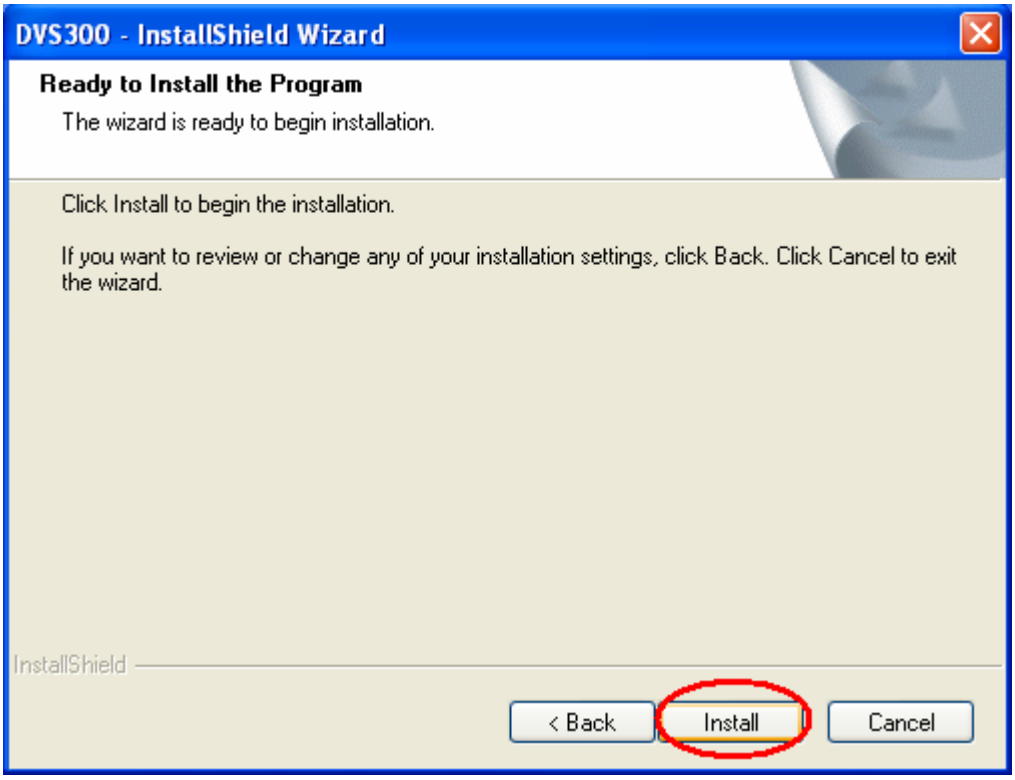
Step 4: Key in your name and company name. Then press the “Next” bottom to continue.



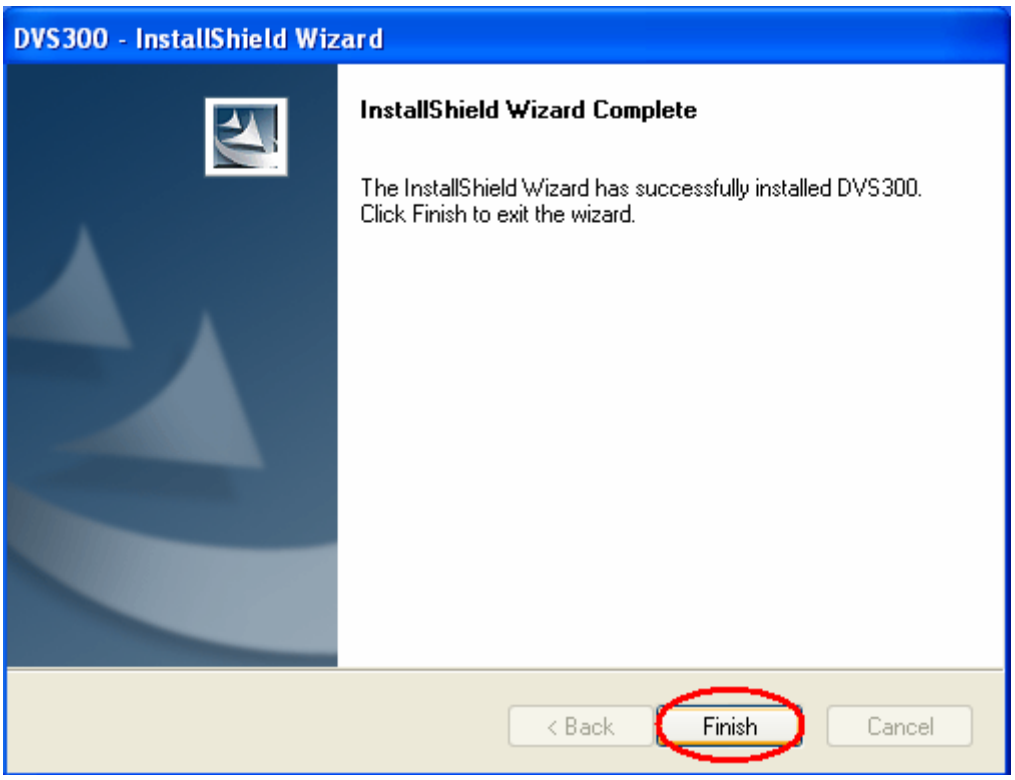
Step 5: Choose the setup type you want and next.



Step 6: Beginning the installation.



Step 7: Finished the installation of DVS-350 demo program.



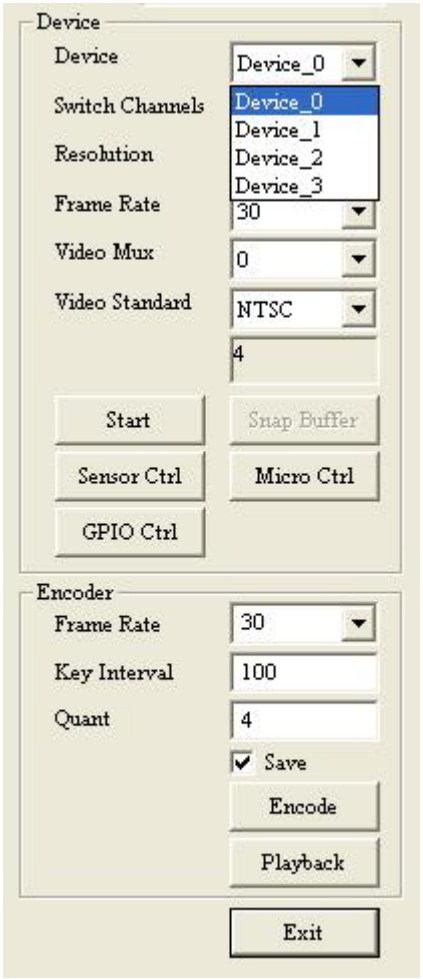
Step 8: There will be a DVS350.exe icon on the desktop. Execute the demo program.

6.3 Demo Program Functionality

Below is the demo program window. The left side panels are the preview windows of video inputs. The right side panels are the function parameter settings.

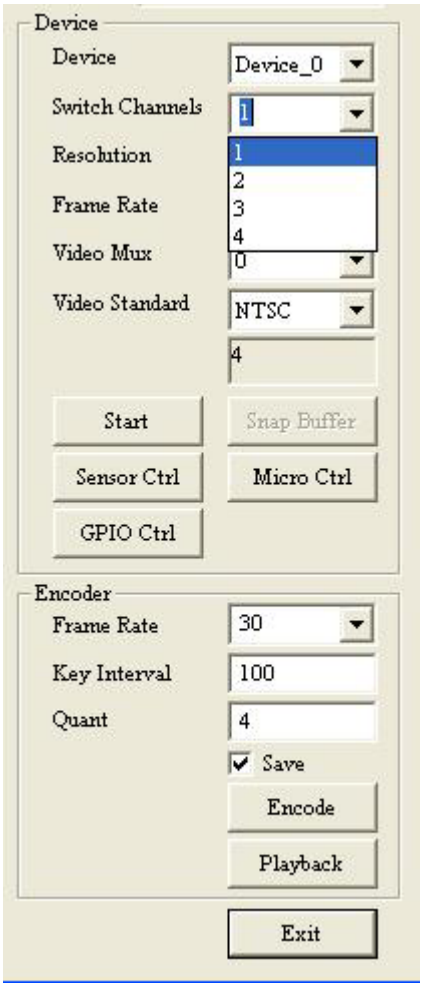
6.3.1 Device

Each device is representative of one Conexant Fusion 878A video capture chip. User can set different parameters to different 878A chip.



6.3.2 Switch Channels

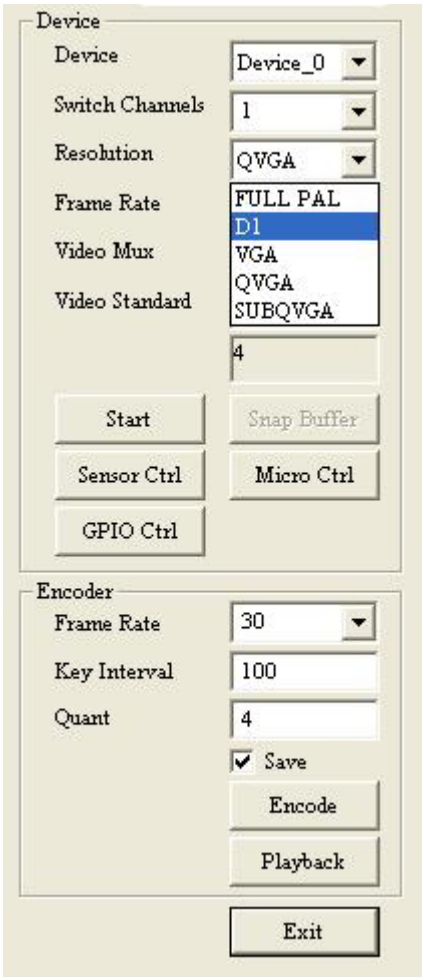
Set the “Switch Channels” to decide how many input for each 878A video chip. Each 878A chip can switch to 4 channel video inputs to share 30/25 frame per second. For more information, please refer to “Chapter 2.2.3 Video Input Connectors” and “ Chapter 7.5.17 DVS300_SetVideoInput”.



6.3.3 Resolution

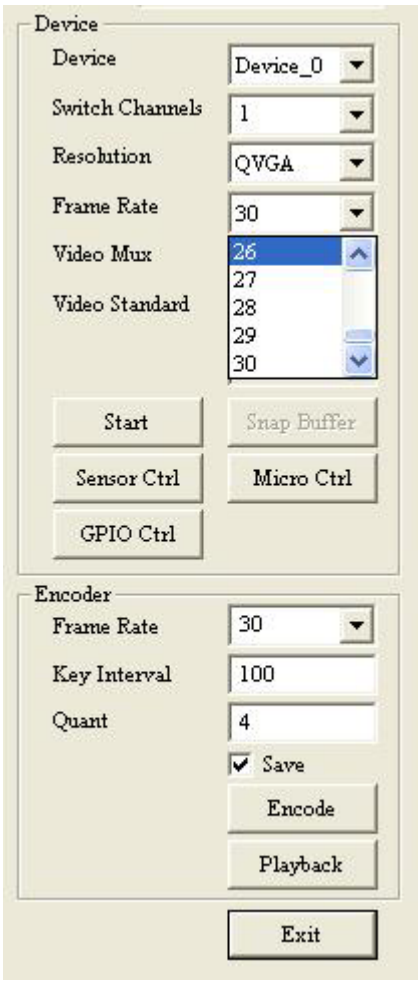
Set the video capturing resolution. Please refer to “Chapter 7.5.14 DVS300_GetResolution” and “Chapter 7.5.15 DVS300_SetResolution”.

Notice: *For the resolution of VGA or D1, the capture video will have the interlace effect on the video image. In other words, there will be lines in the capture image especially when the targeted image is moving. To eliminate this effect, user might need to set the resolution down to 640x240 and use specific algorisms to compensate the image interlace between the scanning even field image and odd field image. For CIF/320x240 resolution, there will be no interlace effect.*



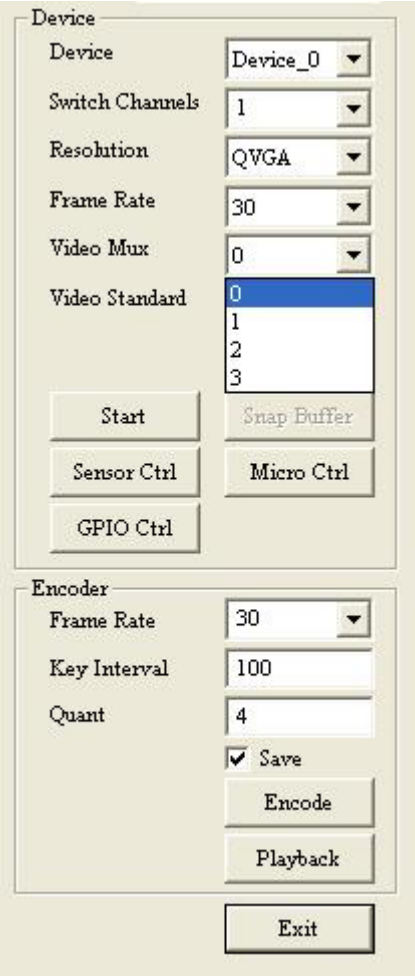
6.3.4 Frame Rate

Set the frame rate for video capturing for specific channel. Please refer to “Chapter 7.5.12 DVS300_GetFrameRate” and “Chapter 7.5.13 DVS300_SetFrameRate”.



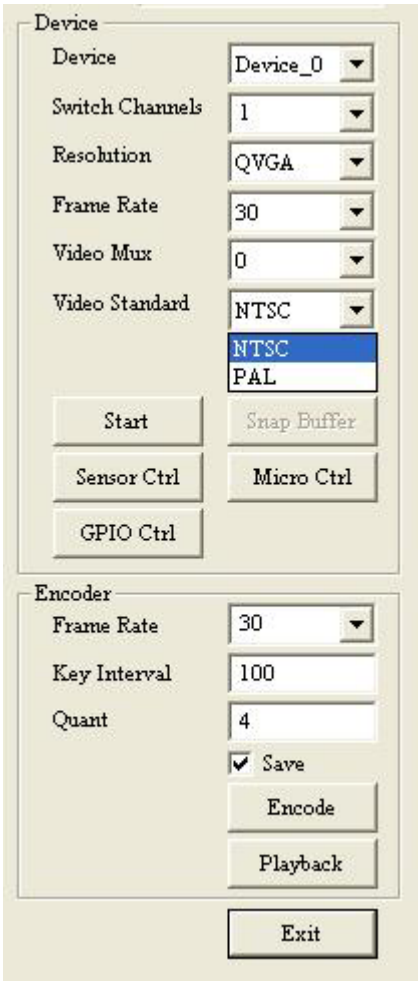
6.3.5 Video Mux

Set the “Video Mux” to specify the video input channel for setting parameter. Please refer to “Chapter 7.5.16 DVS300_GetVideoInput” and “Chapter 7.5.17 DVS300_SetVideoInput”



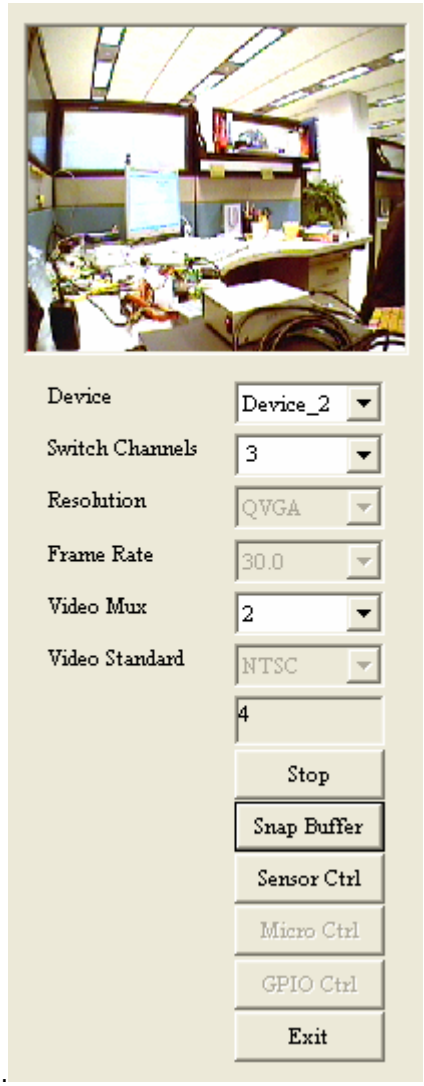
6.3.6 Video Standard

Set the video standard of your cameras. Please refer to “Chapter 7.5.10 DVS300_GetVideoFormat” and “Chapter 7.5.11 DVS300_SetVideoFormat”.



6.3.7 Snap Buffer

Press the “Snap Buffer” to get the image data of specific channel video input. The snap image will be show on the up panel.



6.3.8 Sensor Control

To set the brightness, contrast, hue and saturation of specific channel. Please refer to chapter

7.5.18 DVS300_GetBrightness

7.5.19 DVS300_SetBrightness

7.5.20 DVS300_GetContrast

7.5.21 DVS300_SetContrast

7.5.22 DVS300_GetHue

7.5.23 DVS300_SetHue

7.5.24 DVS300_GetSaturation

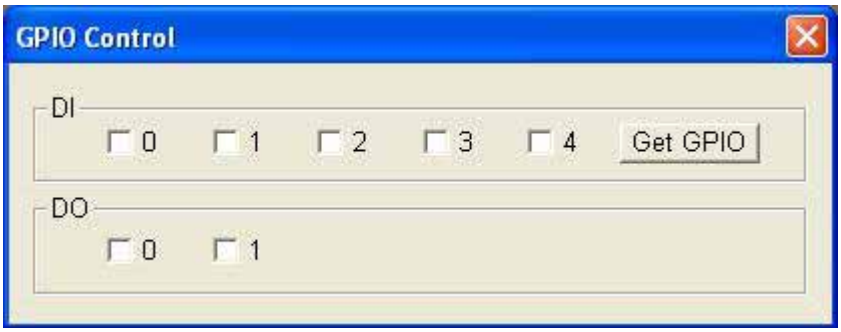
7.5.25 DVS300_SetSaturation

6.3.9 Micro Control

Specify or get the word address(0~127).with a value. Please refer to “Chapter 7.5.29 DVS300_SetEEData” and “Chapter 7.5.28 DVS300_GetEEData”.

6.3.10 GPIO control

To get a specified DI value or to set a specified DO value. Please refer to “Chapter 2.2.4 Isolated DI and Relay DO”, “Chapter 7.5.27 DVS300_GPIOSetData” and “Chapter 7.5.26 DVS300_GPIOGetData”.



CHAPTER 7

Software Function Library

Chapter 7 Software Function Library

7.1 DVS300 Functions Library: DVS300.dll

7.1.1 Data Type Summary

Res	The method returned code

7.1.2 Method Summary

SDK Initialize and close	
DVS300_CreateSDKInstence	Creates SDK instance
DVS300_InitSDK	Initializes all DVS300capture devices
DVS300_CloseSDK	Cleans all instances of capture devices and closes up the SDK.

Capture control	
DVS300_GetNoOfDevices	Gets number of DVP300 Capture Devices
DVS300_Start	Starts video capturing
DVS300_Stop	Stops video capturing
DVS300_GetCapState	Gets capture state
DVS300_SetNewFrameCallback	Sets a callback function to SDK
DVS300_GetCurFrameBuffer	Gets current frame buffer

Capture setting	
DVS300_GetVideoFormat	Gets video input format
DVS300_SetVideoFormat	Sets video input format
DVS300_GetFrameRate	Gets frame rate
DVS300_SetFrameRate	Sets frame rate
DVS300_GetResolution	Gets video resolution
DVS300_SetResolution	Sets video resolution
DVS300_GetVideoInput	Gets video input mux
DVS300_SetVideoInput	Sets video input mux

Sensor Control	
DVS300_GetBrightness	Gets brightness value
DVS300_SetBrightness	Sets brightness value
DVS300_GetContrast	Gets contrast value
DVS300_SetContrast	Sets contrast value
DVS300_GetHue	Gets hue value
DVS300_SetHue	Sets hue value
DVS300_GetSaturation	Gets saturation value
DVS300_SetSaturation	Sets saturation value

GPIO	
DVS300_GPIOGetData	Gets value of specified GPIO pin
DVS300_GPIOSetData	Sets value of specified GPIO pin

Micro Controller	
DVS300_GetEEData	Reads the value at specified EE word address
DVS300_SetEEData	Writes the value at specified EE word address

7.2 DVS300 Encoding Functions Library: DVS300Encoder.dll/Encoder: rmp4.dll

Before using the DVS300 encoding functions library, the “RMP4” codec must be installed to the system. After installing the sample program, the codec will be installed automatically. You can install the codec manually by using the “rmp4.inf” file. Right click on the file, and then click “Install”.

7.2.1 Data Type Summary

EncRes	The method returned code
PSTREAMREADBEGIN	The stream Read Begin function pointer
PSTREAMREADPROC	The Stream Read Process function pointer
PSTREAMREADEND	The Stream Read End function pointer
STREAMREAD_STRUCT	The structure stores the Stream Read callback function pointers

7.2.2 Method Summary

SDK Initialize and close	
DVS300_CreateEncSDKInstence	Creates encoding SDK instance
DVS300_InitSDK	Initializes the SDK
DVS300_CloseSDK	Closes up the SDK
DVS300_InitEncoder	Opens and initializes video encoder
DVS300_CloseEncoder	Closes and release video encoder

Encode control	
DVS300_StartVideoEncode	Starts video encoding
DVS300_VideoEncode	Encodes one video frame
DVS300_StopVideoEncode	Stops video encoding
DVS300_GetState	Gets encoder state
DVS300_CreateAVIFile	Creates an AVI file
DVS300_WriteAVIFile	Writes video data to the AVI file
DVS300_CloseAVIFile	Closes AVI file
DVS300_SetStreamReadCB	Sets the stream read callback functions to SDK

Encode setting	
DVS300_GetVideoQuant	Gets video encoding quant
DVS300_SetVideoQuant	Sets video encoding quant
DVS300_GetVideoFrameRate	Gets video encoding frame rate
DVS300_SetVideoFrameRate	Sets video encoding frame rate
DVS300_GetVideoResolution	Gets video encoding resolution
DVS300_SetVideoResolution	Sets video encoding resolution
DVS300_GetVideoKeyInterval	Gets video encoding key interval
DVS300_SetVideoKeyInterval	Sets video encoding key interval

7.3 DVS 300 Playback Functions Library:

DVS300Player.dll/Decoder: rmp4.dll

Before using the DVS300 playback functions library, the “RMP4” codec must be installed to the system. After installing the sample program, the codec will be installed automatically. You can install the codec manually by using the “rmp4.inf” file. Right click on the file, and then click “Install”.

7.3.1 Data Type Summary

PlayerRes	The method returned code

7.3.2 Method Summary

Playback SDK initialize	
DVS300_CreatePlayerSDKInstence	Creates Playback SDK instance

Playback control	
DVS300_OpenFile	Opens file and initialize player
DVS300_CloseFile	Closes file that has been opened
DVS300_Play	Plays file that has been opened
DVS300_Pause	Pauses or continues
DVS300_Stop	Stops to play file
DVS300_Fast	Plays file with faster speed
DVS300_Slow	Plays file with slower speed
DVS300_PlayStep	Plays by single frame
DVS300_GetStatus	Gets playback state
DVS300_GetCurImage	Gets frame that is rendered
DVS300_RegNotifyMsg	Registers message sent to player when event occurs
DVS300_CheckFileEnd	Checks if file is finished playing

Playback setting	
DVS300_GetVideoResolution	Gets video resolution of file
DVS300_GetFileTime	Gets total file time
DVS300_GetPlayedTime	Gets current file time
DVS300_SetPlayPosition	Locates position of file
DVS300_GetFileTotalFrames	Gets total frame number of file
DVS300_GetPlayedFrames	Gets current frame number of file
DVS300_GetPlayRate	Gets current played rate

7.4 DVS300 Functions Reference

Data Type

7.4.1 Res

Syntax

```
typedef enum tagRes
{
        SUCCEEDED               = 1,
        FAILED                  = 0,
        SDKINITFAILED           = -1,
        PARAMERROR              = -2,
        NODEVICES               = -3,
        NOSAMPLE                = -4,
        DEVICENUMERROR          = -5,
        INPUTERROR              = -6,
        VERIFYHWERROR           = -7
} Res;
```

Description

The method returned code.

7.5 Method

7.5.1 DVS300_CreateSDKInstence

Syntax

int DVS300_CreateSDKInstence(void **pp)

Parameters

pp:	A pointer to the SDK.

Return Value

SUCCEEDED:	Function succeeded.
FAILED:	Function failed.
PARAMERROR:	Parameter error.

Description

This function creates SDK instance.

7.5.2 DVS300_InitSDK

Syntax

int DVS300_InitSDK()

Parameters

None

Return Value

SUCCEEDED:	Function succeeded.
FAILED:	Function failed.
NODEVICES:	No devices found.
VERIFYHWERROR	Verify the hardware error.

Description

This function initializes all DVS300 capture devices in the system. After initializing each device, the capture status would be set as “STOPPED”.

See Also

DVS300_GetNoOfDevices

DVS300_GetCapState

DVS300_CloseSDK

7.5.3 DVS300_CloseSDK

Syntax

int DVS300_CloseSDK(void)

Parameters

None

Return Value

SUCCEEDED:	Function succeeded.
SDKINITFAILED:	SDK not initialized.

Description

This function cleans all instances of capture devices and closes up the SDK.

See Also

DVS300_InitSDK

7.5.4 DVS300_GetNumberOfDevices

Syntax

int DVS300_GetNoOfDevices(int *pNoOfDevs)

Parameters

pNoOfDevs:	A pointer to get number of DVS300 Capture Devices.

Return Value

SUCCEEDED:	Function succeeded.
FAILED:	Function failed.
SDKINITFAILED:	SDK not initialized.

Description

This function gets number of DVS300 Capture Devices in the system. At most 16 channels are available in a DVS300 system.

7.5.5 DVS300_Start

Syntax

int DVS300_Start(int nDevNum, int SwitchingChans, HWND Main, HWND hwndPreview)

Parameters

nDevNum:	Specifies the device number(0~3).
SwitchingChans:	Single video input or switching between video muxes. 0: single channel. 1: two channels (mux0, mux1). 2: three channels (mux0, mux1, mux2). 3: four channels (mux0, mux1, mux2, mux3).
Main:	A main window handle.
hwndPreview:	A windows handle for display area. When the value of this parameter is NULL, the video will not be rendered. (This parameter is only valid, when the “SwitchChans” is zero.)

Return Value

SUCCEEDED:	Function succeeded.
FAILED:	Function failed.
DEVICENUMERROR:	Invalid device number.
SDKINITFAILED:	SDK not initialized.

Description

This function starts video capturing on a specified capture port.

The capture state would be set as “RUNNING” after a successful start. If the channels share frames (i.e. SwitchingChans>0), the video input mux will be set zero.

See Also

DVS300_Stop

DVS300_GetCapState

7.5.6 DVS300_Stop

Syntax

int DVS300_Stop(int nDevNum)

Parameters

nDevNum:	Specifies the device number(0~3).

Return Value

SUCCEEDED:	Function succeeded.
FAILED:	Function failed.
DEVICENUMERROR:	Invalid device number.
SDKINITFAILED:	SDK not initialized.

Description

This function stops video capturing on a specified capture port. The capture state would be set as “STOPPED” after a successful stop.

See Also

DVS300_Start

DVS300_GetCapState

7.5.7 DVS300_GetCapState

Syntax

int DVS300_GetCapState(int nDevNum)

Parameters

nDevNum: Specifies the device number(0~3).

Return Value

DEVICENUMERROR: Invalid device number.

SDKINITFAILED: SDK not initialized.

Description

This function gets capture state of a specified capture port.

```
typedef enum {
   STOPPED          = 1,
   RUNNING          = 2,
   UNINITIALIZED    = -1,
   UNKNOWNSTATE     = -2
} CapState;
```

See Also

DVS300_InitSDK

DVS300_Start

DVS300_Stop

7.5.8 DVS300_GetCurFrameBuffer

Syntax

int DVS300_GetCurFrameBuffer(int nDevNum, long* bufSize, BYTE* buf, int VMux)

Parameters

nDevNum:	Specifies the device number(0~3).
bufSize:	Frame buffer size.
buf:	Frame buffer.
VMux:	Video mux.

Return Value

SUCCEEDED:	Function succeeded.
FAILED:	Function failed.
DEVICENUMERROR:	Invalid device number.
PARAMERROR:	Invalid parameter.
SDKINITFAILED:	SDK not initialized.
NOSAMPLE:	No buffer sample.

Description

This function gets current frame buffer of a specified capture port. Start capturing before the function is called.

See Also

DVS300_Start

7.5.9 DVS300_SetNewFrameCallback

Syntax

int DVS300_SetNewFrameCallback(int nDevNum, int callback)

Parameters

nDevNum:	Specifies the device number(0~3).
callback:	Callback function.
Callback fumction type:	typedef int (*CAPCALLBACK)(int nID, int nDevNum, int VMux, int bufsize, BYTE* buf);
nID:	Single video input ID or the video mux ID. The value of IDs is showed as following: #define ID_NEW_FRAME 37810 #define ID_MUX0_NEW_FRAME 37800 #define ID_MUX1_NEW_FRAME 37801 #define ID_MUX2_NEW_FRAME 37802 #define ID_MUX3_NEW_FRAME 37803
nDevNum:	Specifies the device number(0~3).
VMux:	Specifies the video mux number(0~3).
bufsize:	An integer pointer of the frame buffer size.
buf:	A BYTE pointer of the frame buffer.

Return Value

SUCCEEDED:	Function succeeded.
DEVICENUMERROR:	Invalid device number.
SDKINITFAILED:	SDK not initialized.

Description

This function sets a callback function to SDK. When new frame arrived, messages and frame information will be sent to callback function.

See Also

7.5.10 DVS300_GetVideoFormat

Syntax

int DVS300_GetVideoFormat(int nDevNum, AnalogVideoFormat* vFormat)

Parameters

nDevNum: Specifies the device number(0~3).

Vformat: A pointer to get video format.

```
typedef enum tagAnalogVideoFormat
{
   Video_None          = 0x00000000,
   Video_NTSC_M        = 0x00000001,
   Video_NTSC_M_J      = 0x00000002,
   Video_PAL_B         = 0x00000010,
   Video_PAL_M         = 0x00000200,
   Video_PAL_N         = 0x00000400,
   Video_SECAM_B       = 0x00001000
} AnalogVideoFormat;
```

Return Value

SUCCEEDED: Function succeeded.

FAILED: Function failed.

DEVICENUMERROR: Invalid device number.

PARAMERROR: Invalid parameter.

SDKINITFAILED: SDK not initialized.

Description

This function gets video input format of a specified capture port.

See Also

DVS300_SetVideoFormat

7.5.11 DVS300_SetVideoFormat

Syntax

int DVS300_SetVideoFormat(int nDevNum, AnalogVideoFormat* vFormat)

Parameters

nDevNum: Specifies the port device number(0~3).

Vformat: video format:

```
typedef enum tagAnalogVideoFormat
{
   Video_None        = 0x00000000,
   Video_NTSC_M      = 0x00000001,
   Video_NTSC_M_J    = 0x00000002,
   Video_PAL_B       = 0x00000010,
   Video_PAL_M       = 0x00000200,
   Video_PAL_N       = 0x00000400,
   Video_SECAM_B     = 0x00001000
} AnalogVideoFormat;
```

Return Value

SUCCEEDED: Function succeeded.

FAILED: Function failed.

DEVICENUMERROR: Invalid device number.

SDKINITFAILED: SDK not initialized.

Description

This function sets video input format a specified capture port. This function should be called before “DVS300_Start”.

See Also

DVS300_GetVideoFormat

7.5.12 DVS300_GetFrameRate

Syntax

int DVS300_GetFrameRate(int nDevNum, int *FrameRate)

Parameters

nDevNum:	Specifies the device number(0~3).
FrameRate:	A pointer to get video frame rate(1~30).

Return Value

SUCCEEDED:	Function succeeded.
FAILED:	Function failed.
DEVICENUMERROR:	Invalid device number.
PARAMERROR:	Invalid parameter.
SDKINITFAILED:	SDK not initialized.

Description

This function gets frame rate of a specified capture port.

See Also

DVS300_SetFrameRate

7.5.13 DVS300_SetFrameRate

Syntax

int DVS300_SetFrameRate(int nDevNum , int SwitchingChans, int FrameRate)

Parameters

nDevNum:	Specifies the device number(0~3).
SwitchingChans:	Single video input or switching between video muxes(0~3). 0: single channel. 1: two channels (mux0, mux1). 2: three channels (mux0, mux1, mux2). 3: four channels (mux0, mux1, mux2, mux3).
FrameRate:	A value to set frame rate(1~30). Default value is 30.

Return Value

SUCCEEDED:	Function succeeded.
FAILED:	Function failed.
DEVICENUMERROR:	Invalid device number.
PARAMERROR:	Invalid parameter.
SDKINITFAILED:	SDK not initialized.

Description

This function sets frame rate of a specified capture port. This function should be called before “DVS300_Start”. If the channels share frames (i.e. SwitchingChans>0), the frame rate must be set 30. Otherwise, the function will return PARAMERROR.

See Also

DVS300_GetFrameRate

7.5.14 DVS300_GetResolution

Syntax

int DVS300_GetResolution(int nDevNum, VideoSize *Size)

Parameters

nDevNum: Specifies the device number(0~3).

Size: A pointer to get video resolution.

```
typedef enum
{
   SIZEFULLPAL=0,          //(PAL:768x576)
   SIZED1,                 // (NTSC: 720x480, PAL: 720x576)
   SIZEVGA,                //(640x480)
   SIZEQVGA,               //(320x240)
   SIZESUBQVGA             //(160x120)
} VideoSize;
```

Return Value

SUCCEEDED:	Function succeeded.
FAILED:	Function failed.
DEVICENUMERROR:	Invalid device number.
PARAMERROR:	Invalid parameter.
SDKINITFAILED:	SDK not initialized.

Description

This function gets video resolution of a specified capture port.

See Also

DVS300_SetResolution

7.5.15 DVS300_SetResolution

Syntax

int DVS300_SetResolution(int nDevNum, VideoSize Size)

Parameters

nDevNum: Specifies the device number(0~3).

Size: A value to set video resolution.

```
typedef enum
{
   SIZEFULLPAL=0,          //(PAL:768x576)
   SIZED1=0,               // (NTSC: 720x480, PAL: 720x576)
   SIZEVGA,                //(640x480)
   SIZEQVGA,               //(320x240)
   SIZESUBQVGA             //(160x120)
} VideoSize;
```

Return Value

SUCCEEDED: Function succeeded.

FAILED: Function failed.

DEVICENUMERROR: Invalid device number.

SDKINITFAILED: SDK not initialized.

Description

This function sets video resolution of a specified capture port. This function should be called before “DVS300_Start”.

See Also

DVS300_GetResolution

7.5.16 DVS300_GetVideoInput

Syntax

int DVS300_GetVideoInput(int nDevNum, int* pInput)

Parameters

nDevNum:	Specifies the device number(0~3).
pInput:	A pointer to get video input mux(0~3).

Return Value

SUCCEEDED:	Function succeeded.
FAILED:	Function failed.
DEVICENUMERROR:	Invalid device number.
PARAMERROR:	Invalid parameter.
SDKINITFAILED:	SDK not initialized.

Description

This function gets video input mux of a specified capture port.

It returns “FAILED” when argument “SwitchingChans” of DVS300_Start was set nonzero. And, the video input mux will be set 0 automatically when argument “SwitchingChans” of DVS300_Start was set nonzero.

See Also

DVS300_Start

DVS300_SetVideoInput

7.5.17 DVS300_SetVideoInput

Syntax

int DVS300_SetVideoInput(int nDevNum, int nInput)

Parameters

nDevNum:	Specifies the device number(0~3).
nInput:	A value to set video input mux(0~3).

Return Value

SUCCEEDED:	Function succeeded.
FAILED:	Function failed.
DEVICENUMERROR:	Invalid device number.
PARAMERROR:	Invalid parameter.
SDKINITFAILED:	SDK not initialized.

Description

This function sets video input mux of a specified capture port.

It returns “FAILED” when argument “SwitchingChans” of DVS300_Start was set nonzero. And, the video input mux will be set zero automatically when running DVS300_Start with nonzero “SwitchingChans” argument.

See Also

DVS300_Start

DVS300_GetVideoInput

7.5.18 DVS300_GetBrightness

Syntax

DVS300_GetBrightness(int nDevNum, int nInput, long *lpValue)

Parameters

nDevNum:	Specifies the device number(0~3).
nInput:	Specifies the video input mux(-1~3). This value must be set -1 when no switching channels.
lpValue:	A long pointer to get brightness value(1~100).

Return Value

SUCCEEDED:	Function succeeded.
FAILED:	Function failed.
DEVICENUMERROR:	Invalid device number.
PARAMERROR:	Invalid parameter.
SDKINITFAILED:	SDK not initialized.

Description

This function gets brightness value of a specified capture port.

See Also

DVS300_SetBrightness

7.5.19 DVS300_SetBrightness

Syntax

int DVS300_SetBrightness(int nDevNum , int nInput, long lValue)

Parameters

nDevNum:	Specifies the device number(0~3).
nInput:	Specifies the video input mux(-1~3). This value must be set -1 when no switching channels.
lValue:	A value to set brightness(0~100).

Return Value

SUCCEEDED:	Function succeeded.
FAILED:	Function failed.
DEVICENUMERROR:	Invalid device number.
PARAMERROR:	Invalid parameter.
SDKINITFAILED:	SDK not initialized.

Description

This function sets brightness value of a specified capture port.

See Also

DVS300_GetBrightness

7.5.20 DVS300_GetContrast

Syntax

int DVS300_GetContrast(int nDevNum, int nInput, long *lpValue)

Parameters

nDevNum:	Specifies the device number(0~3).
nInput:	Specifies the video input mux(-1~3). This value must be set -1 when no switching channels.
lpValue:	A long pointer to get contrast value(1~100).

Return Value

SUCCEEDED:	Function succeeded.
FAILED:	Function failed.
DEVICENUMERROR:	Invalid device number.
PARAMERROR:	Invalid parameter.
SDKINITFAILED:	SDK not initialized.

Description

This function gets contrast value of a specified capture port.

See Also

DVS300_SetContrast

7.5.21 DVS300_SetContrast

Syntax

int DVS300_SetContrast(int nDevNum, int nInput, long lValue)

Parameters

nDevNum:	Specifies the device number(0~3).
nInput:	Specifies the video input mux(-1~3). This value must be set -1 when no switching channels.
lValue:	A value to set contrast(0~100).

Return Value

SUCCEEDED:	Function succeeded.
FAILED:	Function failed.
DEVICENUMERROR:	Invalid device.
PARAMERROR:	Invalid parameter.
SDKINITFAILED:	SDK not initialized.

Description

This function sets contrast value of a specified capture port.

See Also

DVS300_GetContrast

7.5.22 DVS300_GetHue

Syntax

int DVS300_GetHue(int nDevNum, int nInput, long *lpValue)

Parameters

nDevNum:	Specifies the device number(0~3).
nInput:	Specifies the video input mux(-1~3). This value must be set -1 when no switching channels.
lpValue:	A long pointer to get hue value(1~100).

Return Value

SUCCEEDED:	Function succeeded.
FAILED:	Function failed.
DEVICENUMERROR:	Invalid device number.
PARAMERROR:	Invalid parameter.
SDKINITFAILED:	SDK not initialized.

Description

This function gets hue value of a specified capture port.

See Also

DVS300_SetHue

7.5.23 DVS300_SetHue

Syntax

int DVS300_SetHue(int nDevNum, int nInput, long lValue)

Parameters

nDevNum:	Specifies the device number(0~3).
nInput:	Specifies the video input mux(-1~3). This value must be set -1 when no switching channels.
lValue:	A value to set hue(0~100).

Return Value

SUCCEEDED:	Function succeeded.
FAILED:	Function failed.
DEVICENUMERROR:	Invalid device number.
PARAMERROR:	Invalid parameter.
SDKINITFAILED:	SDK not initialized.

Description

This function sets hue value of a specified capture port.

See Also

DVS300_GetHue

7.5.24 DVS300_GetSaturation

Syntax

int DVS300_GetSaturation(int nDevNum, int nInput, long *lpValue)

Parameters

nDevNum:	Specifies the device number(0~3).
nInput:	Specifies the video input mux(-1~3). This value must be set -1 when no switching channels.
lpValue:	A long pointer to get saturation value(1~100).

Return Value

SUCCEEDED:	Function succeeded.
FAILED:	Function failed.
DEVICENUMERROR:	Invalid device number.
PARAMERROR:	Invalid parameter.
SDKINITFAILED:	SDK not initialized.

Description

This function gets saturation value of a specified capture port.

See Also

DVS300_SetSaturation

7.5.25 DVS300_SetSaturation

Syntax

int DVS300_SetSaturation(int nDevNum , int nInput, long lValue)

Parameters

nDevNum:	Specifies the device number(0~3).
nInput:	Specifies the video input mux(-1~3). This value must be set -1 when no switching channels.
lValue:	A value to set saturation(0~100).

Return Value

SUCCEEDED:	Function succeeded.
FAILED:	Function failed.
DEVICENUMERROR:	Invalid device number.
PARAMERROR:	Invalid parameter.
SDKINITFAILED:	SDK not initialized.

Description

This function sets saturation value of a specified capture port.

See Also

DVS300_GetSaturation

7.5.26 DVS300_GPIOGetData

Syntax

int DVS300_GPIOGetData(int nDINum, BOOL* pValue)

Parameters

nDINum:	Specifies the digital input number(0~5).
pValue:	A pointer to get the value of the specified digital input.

Return Value

SUCCEEDED:	Function succeeded.
FAILED:	Function failed.
PARAMERROR:	Invalid parameter.
SDKINITFAILED:	SDK not initialized.

Description

This function gets the value of the specified digital input.

See Also

DVS300_GPIOSetData

7.5.27 DVS300_GPIOSetData

Syntax

int DVS300_GPIOSetData(int nDONum, BOOL bValue)

Parameters

nDONum:	Specifies the digital output number(0~1).
bValue:	A value to set the value of the specified digital output.

Return Value

SUCCEEDED:	Function succeeded.
FAILED:	Function failed.
PARAMERROR:	Invalid parameter.
SDKINITFAILED:	SDK not initialized.

Description

This function sets the value of the specified digital output.

See Also

DVS300_GPIOGetData

7.5.28 DVS300_GetEEData

Syntax

int DVS300_GetEEData(BYTE wordAddr, BYTE* pData)

Parameters

wordAddr:	Specifies the word address(0~127).
pData:	A pointer to get byte value stored in EE(0~255).

Return Value

SUCCEEDED:	Function succeeded.
FAILED:	Function failed.
PARAMERROR:	Invalid parameter.
SDKINITFAILED:	SDK not initialized.

Description

This function read the value at specified EE word address.

See Also

DVS300_SetEEData

7.5.29 DVS300_SetEEData

Syntax

int DVS300_SetEEData(BYTE wordAddr, BYTE* pData)

Parameters

wordAddr:	Specifies the word address(0~127).
pData:	A value to set the byte value in EE(0~255).

Return Value

SUCCEEDED:	Function succeeded.
FAILED:	Function failed.
PARAMERROR:	Invalid parameter.
SDKINITFAILED:	SDK not initialized.

Description

This function writes the value at specified EE word address.

See Also

DVS300_GetEEData

7.6 DVS300 Encoding Functions Reference

Data Type

7.6.1 EncRes

Syntax

```
typedef enum tagRes
{
        ENC_SUCCEEDED           = 1,
        ENC_FAILED              = 0,
        ENC_SDKINITFAILED       = -1,
        ENC_ENCINITFAILED       = -2,
        ENC_PARAMERROR          = -3,
        ENC_VERIFYHWERROR       = -4,
        ENC_ENCNUMERROR         = -5,
        ENC_BUFFERFULL          = -6
} EncRes;
```

Description

The method returned code.

7.6.2 PSTREAMREADBEGIN

Syntax

void (*PSTREAMREADBEGIN)(int nEncNum)

Parameters

nEncNum: Specifies the encoder number.

Return Value

None

Description

The pointer to the Stream Read Begin callback function called when begins the video stream read process.

See Also

STREAMREAD_STRUCT

7.6.3 PSTREAMREADPROC

Syntax

void (*PSTREAMREADPROC)(int nEncNum, LPVOID pStreamBuf, long lBufSize, DWORD dwCompFlags)

Parameters

nEncNum:	Specifies the encoder number.
pStreamBuf:	A point to the data buffer stores an encoded video frame.
lBufSize:	Specifies the size of the encoded video frame.
dwCompFlags:	Specifies if this encoded video frame is I-frame. The AVIIF_KEYFRAME value means the frame is I-frame.

#define AVIIF_KEYFRAME 0x00000010L

Return Value

None

Description

The pointer to the Stream Read Process callback function called after every video frame is encoded. User can use this function to get every encoded video frame.

See Also

STREAMREAD_STRUCT

7.6.4 PSTREAMREADEND

Syntax

void (*PSTREAMREADEND)(int nEncNum)

Parameters

nEncNum: Specifies the encoder number.

Return Value

None

Description

The pointer to the Stream Read End callback function called when the video stream read process is finished.

See Also

STREAMREAD_STRUCT

7.6.5 STREAMREAD_STRUCT structure

Syntax

```
typedef struct
{
   void (*PSTREAMREADBEGIN)(int nEncNum);
   void (*PSTREAMREADPROC)(int nEncNum, LPVOID pStreamBuf,
   long lBufSize, DWORD dwCompFlags);
   void (*PSTREAMREADEND)(int nEncNum);
}STREAMREAD_STRUCT;
```

Parameters:

PSTREAMREADBEGIN:	The pointer to the Stream Read Begin callback function called when begins the video stream read process.
PSTREAMREADPROC:	The pointer to the Stream Read Process callback function called after every video frame is encoded.
PSTREAMREADEND:	The pointer to the Stream Read End callback function called when the video stream read process is finished.

Description

This structure stores the Stream Read callback function pointers.

See Also

PSTREAMREADBEGIN

PSTREAMREADPROC

PSTREAMREADEND

DVS300_SetStreamReadCB

7.7 Method

7.7.1 DVS300_CreateEncSDKInstence

Syntax

int DVS300_CreateEncSDKInstence (void **pp)

Parameters

pp:	A pointer to the encoding SDK.

Return Value

ENC_SUCCEEDED:	Function succeeded.
ENC_FAILED:	Function failed.
ENC_PARAMERROR:	Parameter error.

Description

This function creates the encoding SDK instance.

7.7.2 DVS300_InitSDK

Syntax

int DVS300_InitSDK(void)

Parameters

None

Return Value

ENC_SUCCEEDED:	Function succeeded.
ENC_VERIFYHWERROR:	Verify the hardware error.

Description

This function initializes all parameters of the SDK in the system.

See Also

DVS300_CloseSDK

7.7.3 DVS300_CloseSDK

Syntax

int DVS300_CloseSDK(void)

Parameters

None

Return Value

ENC_SUCCEEDED: Function succeeded.

ENC_SDKINITFAILED: SDK does not be initialized successfully.

Description

This function cleans all parameters of the SDK and closes up the SDK.

See Also

DVS300_InitSDK

7.7.4 DVS300_InitEncoder

Syntax

int DVS300_InitEncoder(int nEncNum, int nEncBufSize)

Parameters

nEncNum:	Specifies the encoder number(0~15).
nEncBufSize:	Specifies the encoding buffer size.

Return Value

ENC_SUCCEEDED:	Function succeeded.
ENC_FAILED:	Function failed.
ENC_SDKINITFAILED:	SDK does not be initialized successfully.
ENC_ENCNUMERROR:	Invalid encoder number.

Description

This function opens and initializes the specified video encoder. After initializing the encoder, the encoding state would be set as “ENC_STOPPED”.

See Also

DVS300_CloseEncoder

DVS300_GetState

7.7.5 DVS300_CloseEncoder

Syntax

int DVS300_CloseEncoder(int nEncNum)

Parameters

nEncNum: Specifies the encoder number(0~15).

Return Value

ENC_SUCCEEDED: Function succeeded.

ENC_FAILED: Function failed.

ENC_SDKINITFAILED: SDK does not be initialized successfully.

ENC_ENCNUMERROR: Invalid encoder number.

ENC_ENCINITFAILED: Encoder does not be initialized successfully.

Description

This function closes and releases the specified video encoder. After successfully calling this function, the encoding state would be set as “ENC_UNINITIALIZED”.

See Also

DVS300_InitEncoder

DVS300_GetState

7.7.6 DVS300_StartVideoEncode

Syntax

int DVS300_StartVideoEncode(int nEncNum)

Parameters

nEncNum: Specifies the encoder number(0~15).

Return Value

ENC_SUCCEEDED: Function succeeded.

ENC_FAILED: Function failed.

ENC_SDKINITFAILED: SDK does not be initialized successfully.

ENC_ENCNUMERROR: Invalid encoder number.

ENC_ENCINITFAILED: Encoder does not be initialized successfully.

Description

This function notifies the specified video encoder to prepare to encode the video. The encode state would be set as “ENC_RUNNING” after a successful beginning.

See Also

DVS300_VideoEncode

DVS300_StopVideoEncode

DVS300_GetState

7.7.7 DVS300_VideoEncode

Syntax

int DVS300_VideoEncode(int nEncNum, LPVOID lpInBuf,
int InBufSize, BOOL bKeyFrame)

Parameters

nEncNum:	Specifies the encoder number(0~15).
lpbiIn:	A pointer to the input buffer stores the source video frame.
InBufSize:	Specifies the size of the input buffer.
bKeyFrame:	Specifies if the video frame is encoded as a I-frame.

Return Value

ENC_SUCCEEDED:	Function succeeded.
ENC_FAILED:	Function failed.
ENC_SDKINITFAILED:	SDK does not be initialized successfully.
ENC_ENCNUMERROR:	Invalid encoder number.
ENC_ENCINITFAILED:	Encoder does not be initialized successfully.
ENC_PARAMERROR:	Parameter error.
ENC_BUFFERFULL:	Encoding buffer is full, the video frame can not be written to the buffer.

Description

This function writes the video frame to the encoding buffer to be encoded by the specified encoder.

See Also

DVS300_StartVideoEncode
DVS300_StopVideoEncode

7.7.8 DVS300_StopVideoEncode

Syntax

int DVS300_StopVideoEncode(int nEncNum)

Parameters

nEncNum: Specifies the encoder number(0~15).

Return Value

ENC_SUCCEEDED: Function succeeded.

ENC_FAILED: Function failed.

ENC_SDKINITFAILED: SDK does not be initialized successfully.

ENC_ENCNUMERROR: Invalid encoder number.

ENC_ENCINITFAILED: Encoder does not be initialized successfully.

Description

This function notifies the specified video encoder to stop encoding and releases all relational resources. The encoding state would be set as “ENC_STOPPED” after a successful stop.

See Also

DVS300_StartVideoEncode

DVS300_VideoEncode

DVS300_GetState

7.7.9 DVS300_GetState

Syntax

int DVS300_GetState(int nEncNum)

Parameters

nEncNum: Specifies the encoder number(0~15).

Return Value

ENC_ENCNUMERROR: Invalid encoder number.

Description

This function gets encoding state of a specified video encoder.

```
typedef enum
{
   ENC_STOPPED              = 1,
   ENC_RUNNING              = 2,
   ENC_UNINITIALIZED        = -1,
} EncoderState;
```

See Also

DVS300_InitEncoder

DVS300_CloseEncoder

DVS300_StartVideoEncode

DVS300_StopVideoEncode

7.7.10 DVS300_SetStreamReadCB

Syntax

void DVS300_SetStreamReadCB(STREAMREAD_STRUCT *pStreamRead)

Parameters

pStreamRead:	A pointer to STREAMREAD_STRUCT structure including the pointers to the StreamRead callback functions.

Return Value

None

Description

This function registers the Stream Read callback functions to the SDK.

See Also

STREAMREAD_STRUCT structure

7.7.11 DVS300_GetVideoQuant

Syntax

int DVS300_GetVideoQuant(int nEncNum, int *nQuant)

Parameters

nEncNum:	Specifies the encoder number(0~15).
nQuant:	A pointer to get the video quant(1~31). The default video quality is 4.

Return Value

ENC_SUCCEEDED:	Function succeeded.
ENC_FAILED:	Function failed.
ENC_SDKINITFAILED:	SDK does not be initialized successfully.
ENC_ENCNUMERROR:	Invalid encoder number.
ENC_ENCINITFAILED:	Encoder does not be initialized successfully.

Description

This function gets video quant of the specified video encoder. The lower video quant can get the compressed video with higher quality and bit rate, vice versa.

See Also

DVS300_SetVideoQuant

7.7.12 DVS300_SetVideoQuant

Syntax

int DVS300_SetVideoQuant(int nEncNum, int nQuant)

Parameters

nEncNum:	Specifies the encoder number (0~15).
nQuant:	A value to set the video quant(1~31). The default video quality is 4.

Return Value

ENC_SUCCEEDED:	Function succeeded.
ENC_FAILED:	Function failed.
ENC_SDKINITFAILED:	SDK does not be initialized successfully.
ENC_ENCNUMERROR:	Invalid encoder number.
ENC_ENCINITFAILED:	Encoder does not be initialized successfully.

Description

This function sets video quant of the specified video encoder. The lower video quant can get the compressed video with higher quality and bit rate, vice versa.

See Also

DVS300_GetVideoQuant

7.7.13 DVS300_GetVideoFrameRate

Syntax

int DVS300_GetVideoFrameRate(int nEncNum, int *nFrameRate)

Parameters

nEncNum: Specifies the encoder number (0~15).

nFrameRate: A pointer to get the video encoding frame rate(1~30).

Return Value

ENC_SUCCEEDED: Function succeeded.

ENC_FAILED: Function failed.

ENC_SDKINITFAILED: SDK does not be initialized successfully.

ENC_ENCNUMERROR: Invalid encoder number.

ENC_ENCINITFAILED: Encoder does not be initialized successfully.

Description

This function gets video encoding frame rate of the specified video encoder.

See Also

DVS300_SetVideoFrameRate

7.7.14 DVS300_SetVideoFrameRate

Syntax

int DVS300_SetVideoFrameRate(int nEncNum, int nFrameRate)

Parameters

nEncNum:	Specifies the encoder number (0~15).
nFrameRate:	A value to set the video encoding frame rate(1~30). The default video frame rate is 30.

Return Value

ENC_SUCCEEDED:	Function succeeded.
ENC_FAILED:	Function failed.
ENC_SDKINITFAILED:	SDK does not be initialized successfully.
ENC_ENCNUMERROR:	Invalid encoder number.
ENC_ENCINITFAILED:	Encoder does not be initialized successfully.

Description

This function sets video encoding frame rate of the specified video encoder.

See Also

DVS300_GetVideoFrameRate

7.7.15 DVS300_GetVideoResolution

Syntax

int DVS300_GetVideoResolution(int nEncNum, int *nWidth, int *nHeight)

Parameters

nEncNum: Specifies the encoder number(0~15).

nWidth: A pointer to get the width of the video.

nHeight: A pointer to get the height of the video.

Return Value

ENC_SUCCEEDED: Function succeeded.

ENC_SDKINITFAILED: SDK does not be initialized successfully.

ENC_ENCNUMERROR: Invalid encoder number.

ENC_ENCINITFAILED: Encoder does not be initialized successfully.

Description

This function gets video resolution of the specified video encoder.

See Also

DVS300_SetVideoResolution

7.7.16 DVS300_SetVideoResolution

Syntax

int DVS300_SetVideoResolution(int nEncNum, int nWidth, int nHeight)

Parameters

nEncNum:	Specifies the encoder number(0~15).
nWidth:	A value to set the width of the video. The default width is 320.
nHeight	A value to set the height of the video. The default height is 240.

Return Value

ENC_SUCCEEDED:	Function succeeded.
ENC_FAILED:	Function failed.
ENC_SDKINITFAILED:	SDK does not be initialized successfully.
ENC_ENCNUMERROR:	Invalid encoder number.
ENC_ENCINITFAILED:	Encoder does not be initialized successfully.

Description

This function sets video resolution of the specified video encoder.

See Also

DVS300_GetVideoResolution

7.7.17 DVS300_GetVideoKeyInterval

Syntax

int DVS300_GetVideoKeyInterval(int nEncNum,
int *nKeyInterval)

Parameters

nEncNum:	Specifies the encoder number(0~15).
nKeyInterval:	A pointer to get the interval of the video key frame(1~300).

Return Value

ENC_SUCCEEDED:	Function succeeded.
ENC_FAILED:	Function failed.
ENC_SDKINITFAILED:	SDK does not be initialized successfully.
ENC_ENCNUMERROR:	Invalid encoder number.
ENC_ENCINITFAILED:	Encoder does not be initialized successfully.

Description

This function gets the interval of the video key frame of the specified video encoder.

See Also

DVS300_SetVideoKeyInterval

7.7.18 DVS300_SetVideoKeyInterval

Syntax

int DVS300_SetVideoKeyInterval(int nEncNum, int nKeyInterval)

Parameters

nEncNum: Specifies the encoder number(0~15).

nKeyInterval: A value to set the interval of the video key frame(1~300). The default value is 100.

Return Value

ENC_SUCCEEDED: Function succeeded.

ENC_FAILED: Function failed.

ENC_SDKINITFAILED: SDK does not be initialized successfully.

ENC_ENCNUMERROR: Invalid encoder number.

ENC_ENCINITFAILED: Encoder does not be initialized successfully.

Description

This function sets the interval of the video key frame of the specified video encoder.

See Also

DVS300_GetVideoKeyInterval

7.7.19 DVS300_CreateAVIFile

Syntax

HANDLE DVS300_CreateAVIFile(LPCSTR lpcsFileName, int nWidth, int nHeight, int nFrameRate)

Parameters

lpcsFileName:	Specifies the file name of the AVI file.
nWidth:	Specifies the width of the video.
nHeight:	Specifies the height of the video.
nFrameRate	Specifies the frame rate of the video.

Return Value

If the function succeeds, the file handle is returned. Otherwise, the function returns NULL.

Description

This function creates the AVI file to save the encoded video stream.

See Also

DVS300_WriteAVIFile

DVS300_CloseAVIFile

7.7.20 DVS300_WriteAVIFile

Syntax

int DVS300_WriteAVIFile(HANDLE hAVIFile, LPVOID lpStreamBuf, long lBufSize, DWORD dwCompFlags)

Parameters

hAVIFile:	Specifies the AVI file handle.
lpStreamBuf:	A pointer to the video stream data buffer written into the file.
lBufSize:	Specifies the size of the video stream data buffer.
dwCompFlags:	Flag associated with this data. The AVIIF_KEYFRAME flag is defined to indicate this data does not rely on preceding data in the file.

#define AVIIF_KEYFRAME 0x00000010L

Return Value

ENC_SUCCEEDED:	Function succeeded.
ENC_FAILED:	Function failed.
ENC_SDKINITFAILED:	SDK does not be initialized successfully.

Description

This function writes the video stream data into the specified AVI file.

See Also

DVS300_CreateAVIFile

DVS300_CloseAVIFile

7.7.21 DVS300_CloseAVIFile

Syntax

int DVS300_CloseAVIFile(HANDLE hAVIFile)

Parameters

hAVIFile: Specifies the AVI file handle.

Return Value

ENC_SUCCEEDED: Function succeeded.

ENC_FAILED: Function failed.

ENC_SDKINITFAILED: SDK does not be initialized successfully.

Description

This function closes the specified AVI file.

See Also

DVS300_CreateAVIFile

DVS300_WriteAVIFile

7.8 Playback Functions Reference

Data Type

7.8.1 PlayerRes

Syntax

```
typedef enum tagRes
{
        PLAYER_SUCCEEDED            = 1,
        PLAYER_FAILED               = 0,
        PLAYER_SDKINITFAILED        = -1,
        PLAYER_PARAMERROR           = -2,
} PlayerRes;
```

Description

The method returned code.

7.9 Method

7.9.1 DVS300_CreatePlayerSDKInstence

Syntax

int DVS300_CreatePlayerSDKInstence(void **pp)

Parameters

pp:	A pointer to the playback SDK.

Return Value

PLAYER_SUCCEEDED:	Function succeeded.
PLAYER_FAILED:	Function failed.
PLAYER_PARAMERROR:	Parameter error.

Description

This function creates playback SDK instance.

7.9.2 DVS300_OpenFile

Syntax

int DVS300_OpenFile(LPCSTR lpcsFileName)

Parameters

lpcsFileName: Specifies the file name of the source video file.

Return Value

PLAYER_SUCCEEDED: Function succeeded.

PLAYER_FAILED: Function failed.

Description

This function opens the source video file and initializes the video player. The playback status would be set as “PLAYER_STOPPED” after successful open.

See Also

DVS300_CloseFile

DVS300_GetStatus

7.9.3 DVS300_CloseFile

Syntax

int DVS300_CloseFile()

Parameters

None.

Return Value

PLAYER_SUCCEEDED: Function succeeded.

PLAYER_FAILED: Function failed.

Description

This function closes the source video file and free resources allocated for video player. The playback status would be set as “PLAYER_NOTOPENED” after successful close.

See Also

DVS300_OpenFile

DVS300_GetStatus

7.9.4 DVS300_Play

Syntax

int DVS300_Play(HWND hwndApp, BOOL bAutoResizeWnd)

Parameters

hwndApp: A windows handle for display area.

bAutoResizeWnd: Specifies if the display area is resized automatically according to the video resolution.

Return Value

PLAYER_SUCCEEDED: Function succeeded.

PLAYER_FAILED: Function failed.

Description

This function plays the opened file. The playback status would be set as “PLAYER_PLAYING” after successfully play.

See Also

DVS300_Pause

DVS300_Stop

DVS300_GetStatus

7.9.5 DVS300_Pause

Syntax

int DVS300_Pause()

Parameters

None.

Return Value

PLAYER_SUCCEEDED: Function succeeded.

PLAYER_FAILED: Function failed.

Description

This function pauses or continues the file that has been opened. The playback status would be set as “PLAYER_PAUSED” after successful pause.

See Also

DVS300_Play

DVS300_Stop

DVS300_GetStatus

7.9.6 DVS300_Stop

Syntax

int DVS300_Stop()

Parameters

None.

Return Value

PLAYER_SUCCEEDED: Function succeeded.

PLAYER_FAILED: Function failed.

Description

This function stops the file that is playing. The playback status would be set as “PLAYER_STOPPED” after successful stop.

See Also

DVS300_Play

DVS300_Pause

DVS300_GetStatus

7.9.7 DVS300_Fast

Syntax

int DVS300_Fast()

Parameters

None.

Return Value

PLAYER_SUCCEEDED: Function succeeded.

PLAYER_FAILED: Function failed.

Description

This function speeds up the current play speed by one time, two times at most. The playback status would be set as “PLAYER_PLAYING” after successful speed up.

See Also

DVS300_Pause

DVS300_Stop

DVS300_Slow

DVS300_GetStatus

7.9.8 DVS300_Slow

Syntax

int DVS300_Slow()

Parameters

None.

Return Value

PLAYER_SUCCEEDED: Function succeeded.

PLAYER_FAILED: Function failed.

Description

This function speeds down the current play speed by one time, two times at most. The playback status would be set as “PLAYER_PLAYING” after successfully speed down.

See Also

DVS300_Pause

DVS300_Stop

DVS300_Fast

DVS300_GetStatus

7.9.9 DVS300_PlayStep

Syntax

int DVS300_PlayStep()

Parameters

None.

Return Value

PLAYER_SUCCEEDED: Function succeeded.

PLAYER_FAILED: Function failed.

Description

This function makes the video to step forward one frame. The playback status would be set as “PLAYER_PAUSED” after successful single step.

See Also

DVS300_Pause

DVS300_Stop

DVS300_GetStatus

7.9.10 DVS300_GetStatus

Syntax

int DVS300_GetStatus ()

Parameters

None

Return Value

PLAYER_SUCCEEDED: Function succeeded.

PLAYER_FAILED: Function failed.

Description

This function gets playback status.

```
typedef enum tagPlayerStatus
{
   PLAYER_NOTOPENED = 0,
   PLAYER_OPENED    = 1,
   PLAYER_PLAYING   = 2,
   PLAYER_STOPPED   = 3,
   PLAYER_PAUSED    = 4
} PlayerStatus;
```

See Also

DVS300_OpenFile

DVS300_CloseFile

DVS300_Play

DVS300_Pause

DVS300_Stop

DVS300_Fast

DVS300_Slow

DVS300_PlayStep

7.9.11 DVS300_GetCurImage

Syntax

int DVS300_GetCurImage(LPBYTE *lpImage,
long *pBufSize)

Parameters

lpImage:	A pointer to a image buffer.
pBufSize:	A long pointer to receive the returned image buffer size.

Return Value

PLAYER_SUCCEEDED:	Function succeeded.
PLAYER_FAILED:	Function failed.

Description

This function gets current played image.

See Also

7.9.12 DVS300_RegNotifyMsg

Syntax

int DVS300_RegNotifyMsg(HWND hWnd, UINT nMsg)

Parameters

hWnd:	Specifies the handle of the window receiving this message.
nMsg:	Specifies the user-define message. When this message is received, it means some event of the playback occur such as the file playing is end.

Return Value

PLAYER_SUCCEEDED:	Function succeeded.
PLAYER_FAILED:	Function failed.

Description

This function registers a user-define message. When an event of the playback occurs, this message will be sent to the specified window. This function must be called after “DVS300_OpenFile” function.

See Also

DVS300_CheckFileEnd

7.9.13 DVS300_CheckFileEnd

Syntax

BOOL DVS300_CheckFileEnd ()

Parameters

None

Return Value

If the event that the end of file is detected, this function returns TRUE. Otherwise, the function returns FALSE.

Description

This function checks if the player reach end of file.

See Also

DVS300_RegNotifyMsg

7.9.14 DVS300_GetVideoResolution

Syntax

int DVS300_GetVideoResolution(int *nWidth, int *nHeight)

Parameters

nWidth:	An integer pointer to get the width of the video.
nHeight:	An integer pointer to get the height of the video.

Return Value

PLAYER_SUCCEEDED:	Function succeeded.
PLAYER_FAILED:	Function failed.

Description

This function gets width and the height of the video.

See Also

7.9.15 DVS300_GetPlayRate

Syntax

double DVS300_GetPlayRate()

Parameters

None

Return Value

If the function succeeded, the playback ratio is returned. Otherwise, the function returns 0.

Description

This function retrieves the playback rate.

See Also

DVS300_Play

DVS300_Fast

DVS300_Slow

7.9.16 DVS300_GetFileTime

Syntax

double DVS300_GetFileTime()

Parameters

None

Return Value

If the function succeeded, the total file time is returned. Otherwise, the function returns 0.

Description

This function retrieves total file time in seconds.

See Also

DVS300_GetPlayedTime

DVS300_SetPlayPosition

7.9.17 DVS300_GetPlayedTime

Syntax

double DVS300_GetPlayedTime()

Parameters

None

Return Value

If the function succeeded, the current file time is returned. Otherwise, the function returns 0.

Description

This function retrieves current file time in seconds.

See Also

DVS300_GetFileTime

DVS300_SetPlayPosition

7.9.18 DVS300_SetPlayPosition

Syntax

int DVS300_SetPlayPosition (double dTime)

Parameters

dTime: Specifies the file time in seconds.

Return Value

PLAYER_SUCCEEDED: Function succeeded.

PLAYER_FAILED: Function failed.

Description

This function seeks the file to the specified file time.

See Also

DVS300_GetFileTime

DVS300_GetPlayedTime

7.9.19 DVS300_GetFileTotalFrames

Syntax

LONGLONG DVS300_GetFileTotalFrames()

Parameters

None

Return Value

If the function succeeded, the total number of the frame in the file is returned. Otherwise, the function returns 0.

Description

This function retrieves total number of the frame in the file.

See Also

DVS300_GetPlayedFrames

7.9.20 DVS300_GetPlayedFrames

Syntax

LONGLONG DVS300_GetPlayedFrames()

Parameters

None

Return Value

If the function succeeded, the current frame number of the file is returned.

Otherwise, the function returns 0.

Description

This function retrieves current frame number of the file.

See Also

DVS300_GetFileTotalFrames

1.•This transmitter must not be co-located or operating in conjunction with any other antenna or transmitter.
2.•This equipment complies with FCC RF radiation exposure limits set forth for an uncontrolled
environment. This equipment should be installed and operated with a minimum distance of 20 centimeters
between the radiator and your body.

You are cautioned that changes or modifications not expressly approved by the party responsible for
compliance could void your authority to operate the equipment.