Atop Technologies, Inc.

SE59XX-SDK Family
Software Development Kit

User Manual

V1.2

May 3rd, 2018

This PDF Document contains internal hyperlinks for ease of navigation.
For example, click on any item listed in the **Table of Contents** to go to that page.

Published by:

Atop Technologies, Inc.

2F, No. 146, Sec. 1, Tung-Hsing Rd,
30261 Chupei City,
Hsinchu County
Taiwan, R.O.C.

Tel: +886-3-550-8137
Fax: +886-3-550-8131
sales@atop.com.tw
www.atoponline.com
www.atop.com.tw

Important Announcement

Disclaimer

We have checked the contents of this manual for agreement with the hardware and the software described. Since deviations cannot be precluded entirely, we cannot guarantee full agreement. However, the data in this manual is reviewed regularly and any necessary corrections will be included in subsequent editions.
Suggestions for improvement are welcome.
All other product's names referenced herein are registered trademarks of their respective companies.

Documentation Control

Author:	Carlos Hsu, Willy Lin, Matteo Tabarelli
Revision:	1.1
Revision History:	Released
Creation Date:	5 February 2018
Last Revision Date:	4 May 2018
Product Reference:	SE5901 (SDK), SE5901B (SDK), SE5904D (SDK), SE5908 (SDK), SE5916 (SDK), SE5900A (SDK), SE5908A (SDK), SE5901A (SDK)
Document Status:	Released

Table of Contents

Table of Figures

List of Tables

1 Preface

1.1 Purpose of the Manual

This manual supports you in understanding the software SDK architecture of ATOP's SE59XX Series and should be a reference guide for application development on this platform.

1.2 Who Should Use This User Manual

This manual is to be used by qualified programmers, network personnel or support technicians who are familiar with network operations and C Language programming. For any related problems, please contact your local distributor. If they are unable to assist you, please redirect your inquiries to www.atop.com.tw or www.atoponline.com.

1.3 Supported Platform

This manual is designed for the SE5901, SE5901B, SE5904D, SE5908, SE5916, SE5900A, SE5908A, and SE5916A Industrial Serial and Ethernet controller and that model only.

1.4 Warranty Period

ATOP provides a **5-year limited warranty** for SE59XX Series.

2 Introduction to Atop SDK

2.1 Overview of SE59XX-SDK development environment

Notice: Please upgrade to the Firmware version on which this SDK document is based.

Figure 2.1 shows the whole architecture of SE59XX SDK. Three types of Applications are provided in user's layer:

1) ATOP applications: providing multiple sample SDK programs to hardware devices
2) ATOP utility: providing firmware upgrade, network settings and storage mounting tools
3) Third-party : providing 3rd parties software required such as SNMP / Apache / SQLite

In Kernel Layer, Linux 3.14.26 is customized to provide complete networking protocols.
In Driver Layer, device drivers for all Industrial communication interfaces are provided.
In hardware Layer, Customized ARM Cortex-A8 platform and Atop FPGA management core are provided.

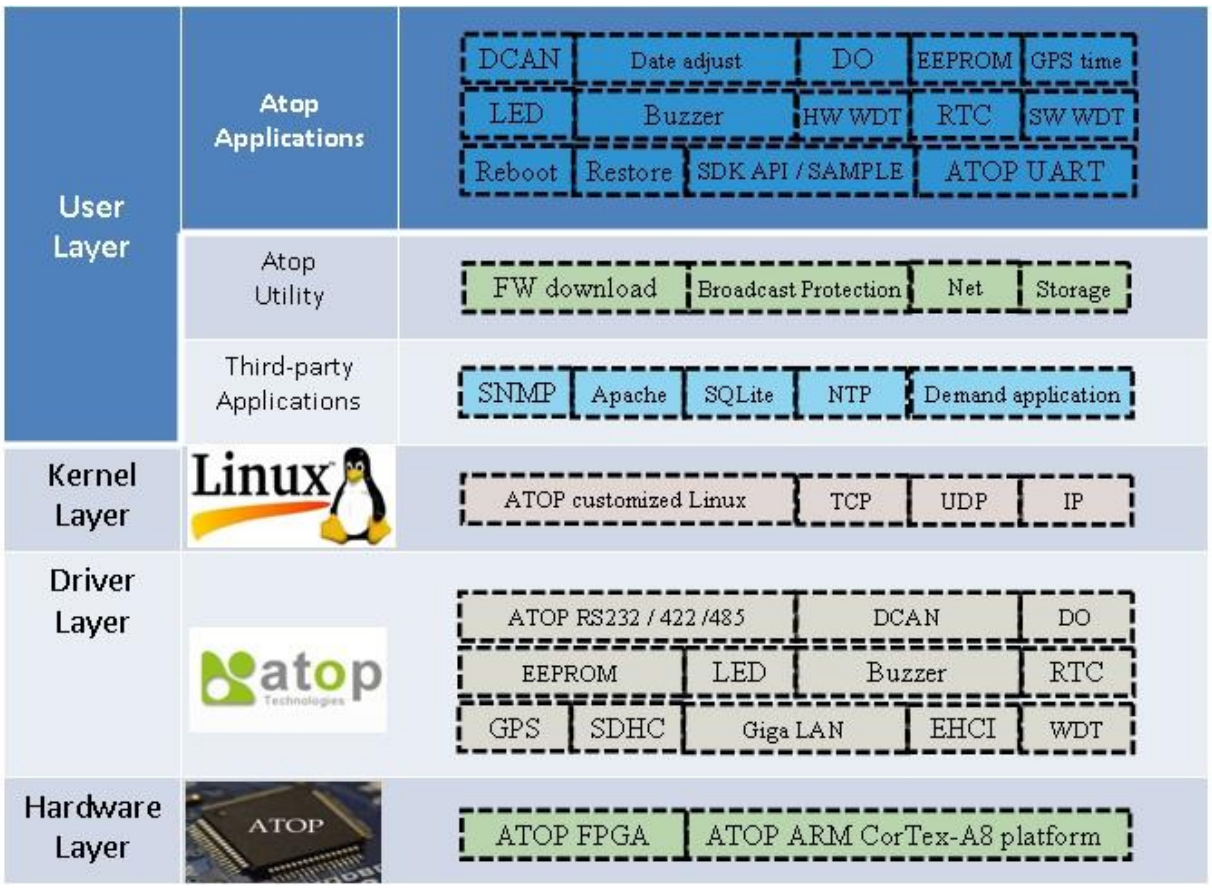


Figure 2.1 Architecture of SE5904D SDK

2.2 Description of SDK Folders

Extract sdk_release_YYYYMMDD.tar.bz2 and refer SDK_Release/ folder (please note that YYYY corresponds to the release year, MM to the release month and DD to the release Day).

There are 4 sub-folders:

- build: this folder includes build done firmware and merge utilities.
- filesystem: this folder includes root file system and bootup script.
- software: this folder includes ATOP library, sample code and header file.
- 3rdparty: 3rd party utilities

The followings are the list of document in "3rdparty" folder:

Table 2.1 Content of 3rd party folder

Folder Name	Description
Busybox-1.23.1	Busybox source
c-ares	C library for asynchronous DNS requests
dhcp-4.1-esv-R13	IPv6 dhcp utilities
email-3.1.3	E-mail utility
gmp-6.1.2	gmp-6utility – for arbitrary precision arithmetic
Hostap	user space daemon for access point and authentication servers.
i2c-tools-3.1.2	I2C tools to manage I2C Bus
iniparser	Ini file parser library
iptables-1.6.1	Tool to manage IP tables
libmodbus-3.1.2	Modbus stack
libnl-3.2.25	libnl suite is a collection of libraries
libpcap-1.7.4	a portable C/C++ library for network traffic capture
libuuid	to generate unique ident for obj to be accessible beyond local system
monit-5.18	Daemon monitor utility
mosquitto-1.4.14	MQTT stack
ncftp-3.2.5	FTP utility
openssl-1.0.2	Openssl library
rtl8192EU_linux_v4.4.1.1	Wi-Fi dongle driver.
strongswan-5.5.2	IPsec VPN
ucarp-1.5.2	allows 2 host share common virtual IP to provide automatic failover
wireless_tools.29	Wifi tools
zlib-1.2.8	Zip library

The followings are the list of application programs in "software" folder:

Table 2.2 Content of Software folder

Folder Name	Description
include	Reference header file
atop_library	ATOP library
atop_application	Sample code

libatop.so.1.0.0	ATOP library binary

The followings are the list of application programs in "filesystem" folder :

Table 2.3 List of programs in filesystem folder

Folder Name	Description
etc	Bootup script, no need to modify under this folder
rootfs.tar.bz2	Pre-build root file system.

The followings are the list of application programs in "build" folder :

Table 2.4 Content of build folder

Folder Name	Description
Image.dld	Build done FW image.
initrd.uboot	Root file system package
composer	Merge image utility
u-boot.bin	Bootloader image for rescuing device
u-boot.dld	Bootloader image for rescuing device
zlimage	Linux kernel raw image

2.3 Firmware upgrade

There are two ways to upgrade the firmware on the SE59XX platform:

2.3.1 Use boot-loader update via console port

Prepare a Debug Cable (RJ45 to Serial) and a CAT5E Ethernet cable. Then, follow below figure to connect the Debug port to PC's COM and CAT5E cable to connect to the Device's LAN1 Ethernet port to any Host PC's Ethernet port.

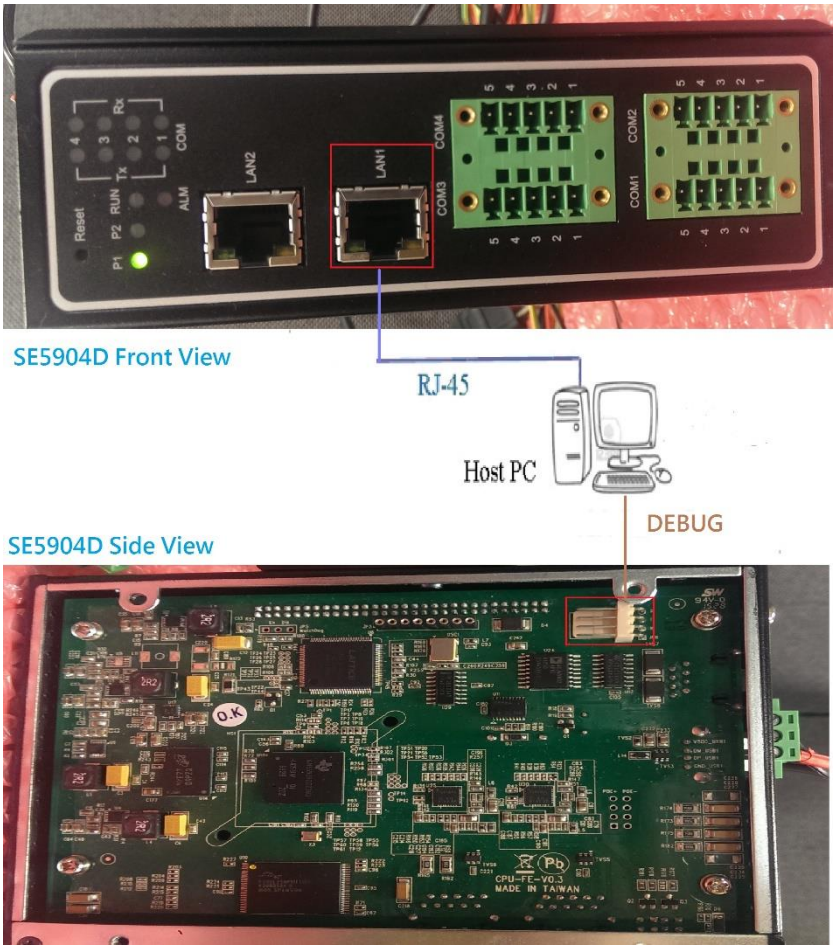


Figure 2.2 Console firmware update- connections

On your PC, run Windows' "Super Terminal" setup COM port parameters as follows:

- Port: the connected COM port
- Baud Rate: 115200 bps
- Parity: none
- Data: 8 data bits
- Stop: 1 stop bit
- Flow control: none

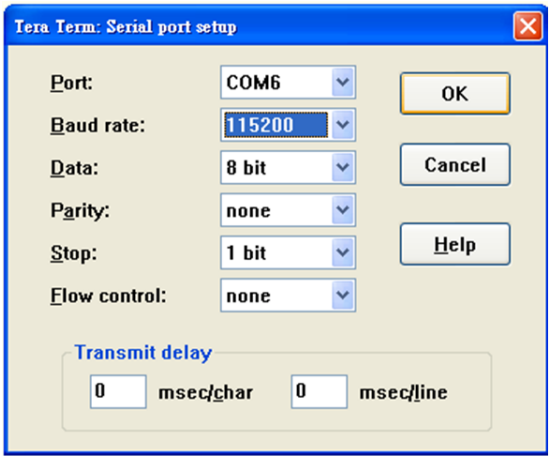


Figure 2.3 COM port Parameters for Console Firmware update

With this method, TFTP protocol is used. The TFTP client is already set-up and running inside the SE59XX platform. Thus, the user needs to execute TFTP server in Windows. An open source version is available for download and can be found as “tftpd32”. Screenshot below shows "tftpd32.exe" after running the application.

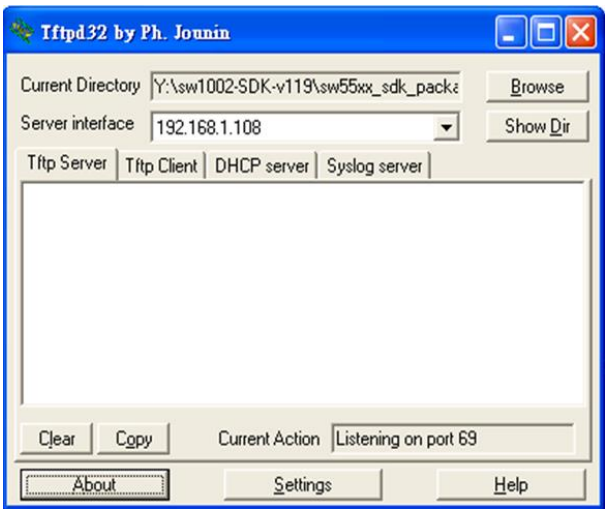


Figure 2.4 TFTPD32 appearance after execution

Now, setup the IP address of the TFTP server. The current folder is the one where "tftpd32.exe" is located. After executing TFTP server, reboot the target SE59XX platform and press the Escape (“Esc”) key immediately. A boot-loader menu will be shown as Figure 2.5.

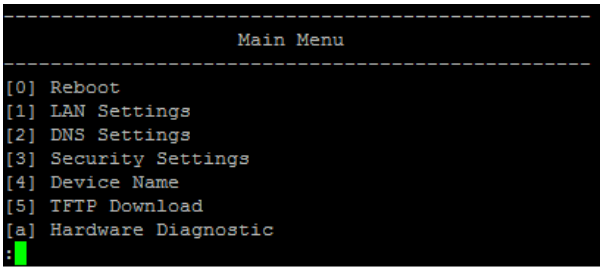


Figure 2.5 SE5904D Boot bader menu

Select item 1 to enter "LAN Setting" menu as Figure 2.6, and setup IP/Netmask/Gateway of LAN1 as Figure 2.7

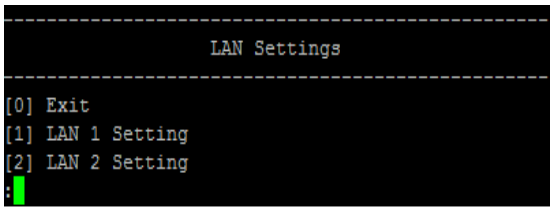


Figure 2.6 LAN Settings

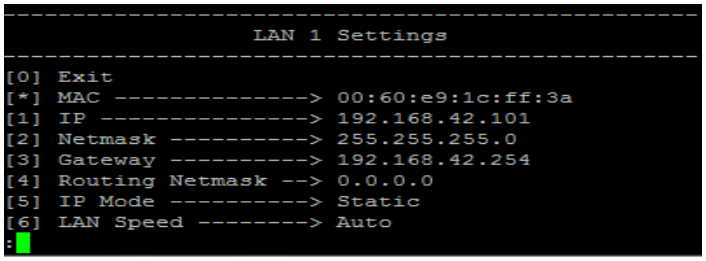


Figure 2.7 LAN1 settings

Enter 0 to exit to upper layer menu and select 5 to enter the "TFTP Download" menu, then select 1 to setup TFTP **server IP as Figure 2.8**

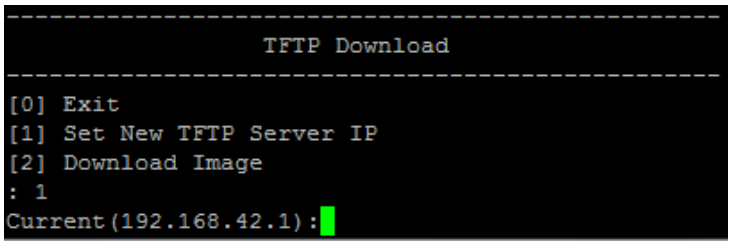


Figure 2.8 TFTP download menu

After the setup of the server IP is completed, select 2 to download the firmware image.

Note: the extension of the firmware should be .dld

2.3.2 Use Device Manager or Device Management Utility

Please use a CAT5E cable to connect SE59XX to a PC running Windows where ATOP Device Management utility is already installed. To install Device Management Utility, please download the latest release from ATOP Website and follow its dedicated user manual for the installation.

The device doesn't have necessarily to be directly connected to the PC, as long as it is inside the same LAN. Atop Management Utility will scan the whole network automatically.

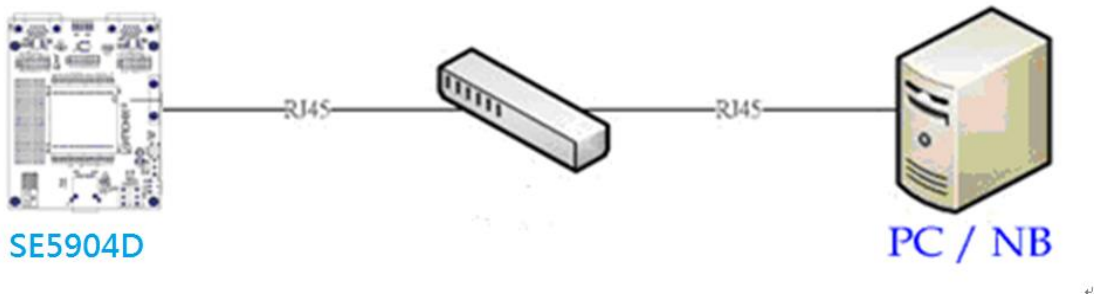


Figure 2.9 SE59XX connection scheme (example on SE5904D)

Now, please power on the device and run Atop's Device Management Utility from your Host PC. Once the device is running, the utility will list all devices found. If the device doesn't show up, push the leftmost button (Rescan function). Once identified, select the device by mouse left button and select "Firmware" >> "Download Firmware" as per Figure 2.10.

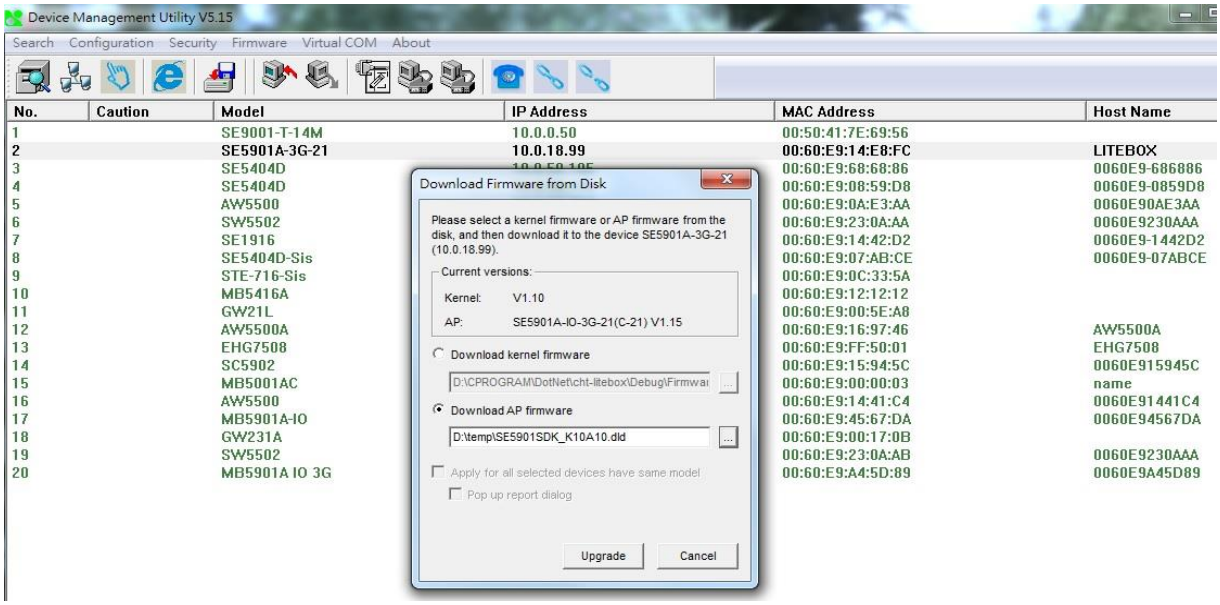


Figure 2.10 Firmware update prompt

Select the firmware (Kernel or AP) from this dialog and select the upgraded file as Figure 2.11. Then, click on the "**Upgrade**" button to upgrade the firmware selected.

Note: This example is made with SE5901A. All other models of SE59XX family share the same method.

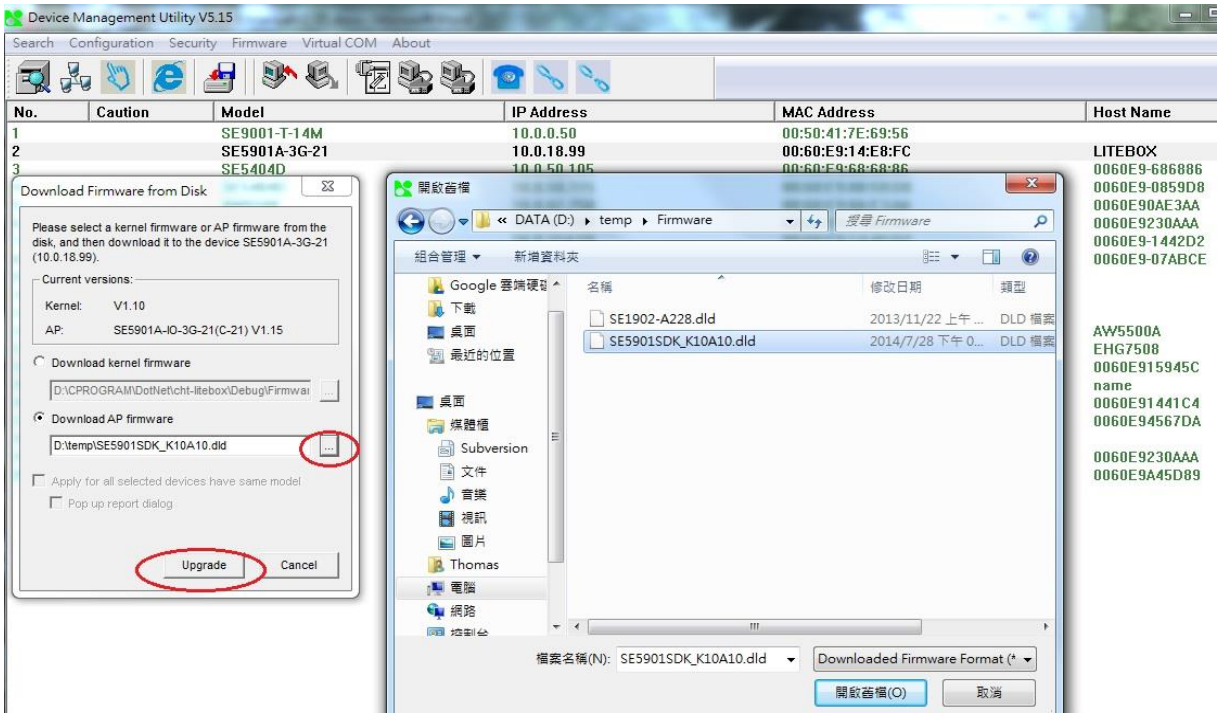


Figure 2.11 Firmware selection

Note that the extension file name of upgraded firmware should be .dld

2.4 Verify current firmware version

There are two methods to verify the firmware version:

1) Use a debug line as per Paragraph 2.3.1 above to connect console port of the device. After bootup, type "atop_show_ver" in the console command line to check current version as Figure 1-13 shown. The red rectangle shows information of boot-bader (V1.00), Kernel (V1.00) and AP (V1.00) version number.

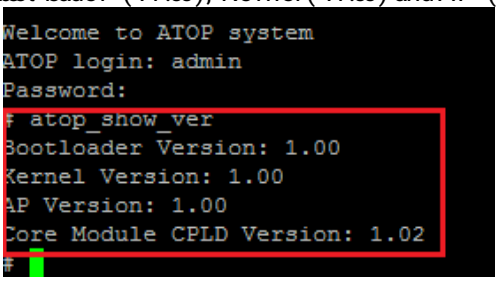


Figure 2.12 Firmware version - Console

2) Use Device Manager or Device Management Utility (Serial Manager) to check version number as per Figure 2.13. (Device Manager is currently supported to Simplified Chinese release)

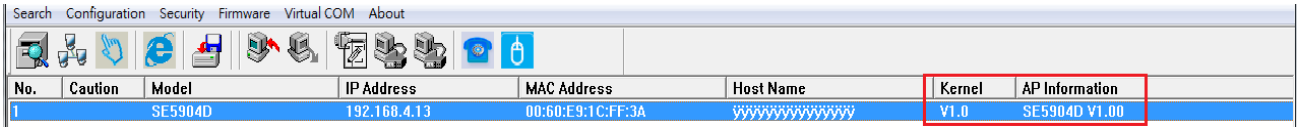


Figure 2.13 Firmware version in Device Management Utility (English)

2.5 Installing a Cross-Compiler

1) Copy ti-am335x-linux-devkit-08.00.00.00.tar.gz to /opt folder and extract it. Be sure that you have and use the root account to do it. This user manual is made with this version. If a newer, stable version is available, the SDK package will include it.

```
tar zxf ./ti-am335x-linux-devkit-08.00.00.00.tar.gz /opt
```

2) Add these environment variable

```
export ARCH=arm
export PATH=/opt/ti-am335x-linux-devkit-08.00.00.00/bin:$PATH
export CROSS_COMPILE=arm-linux-gnueabihf-
```

2.6 Compiling Procedure for Atop Applications

To compile ATOP application, in SDK root folder, please enter

`make clean`	clear all .o object and executable files
`make release platform=am335x_v8`	compile and link the source code

After build done, you can find your image under build folder be named Image.dld.

2.7 Compiling new Applications

1) Put the source code under ./software/atop_application/utils/<YOUR_APP_FOLDER> folder. <YOUR_APP_FOLDER> is a name chosen by yourself. (such as "Test")
2) Follow Paragraph 2.6 above to build your application or modify "Makefile" following the /atop_sdk.

2.8 Download new Applications to the device

New applications can be downloaded in two ways:

2.8.1 Using TFTP protocol

- Please execute tftpd32.exe in the remote PC and modify target folder and IP address as **Figure 2.8**
- Login into the target device (under Linux console) and enter:

```
tftp -gr YOUR_APP_NAME YOUR_TFTP_SERVER_IP
```

Remember to use "chmod" command to modify the access attributes of these files. If transmission failed, please check the networking connection.

2.8.2 Using FTP protocol

- Setup or read FTP account and password from Atop boot-bader menu as per image below.

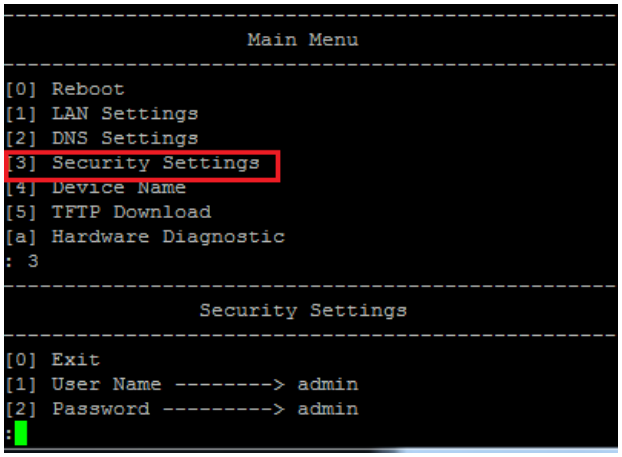


Figure 2.14 FTP access credentials

- Login the Linux system in order to make sure the network connection is fine.
  Use any 3rd party ftp software to transfer the files. For example, use FileZilla as

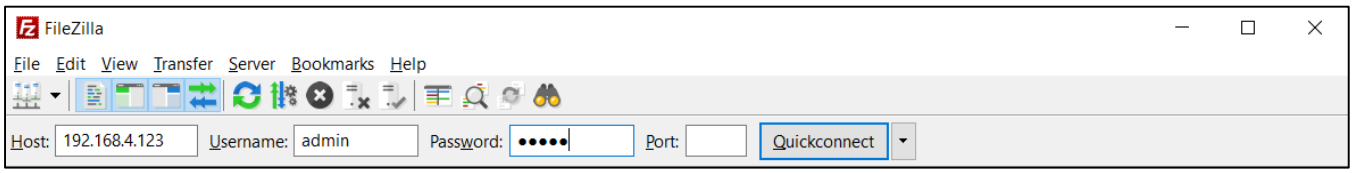


Figure 2.15 FTP Download with FileZilla

- Input FTP account / password of SE59XX and login to the FTP server.

Note: Make sure the binary mode to be set during the transmission.

Remember to use "chmod" command to modify the access attribute of these files. If transmission failed, please check the networking connection okay or not between SE59XX platform and Host PC.

2.9 Login or Remote Login to the device

2.9.1 Remote Login

1) Setup or read FTP account and password from Atop boot-loader menu as Figure 1-16
2) Use any tools supporting the telnet protocol such as "ssh" inside of Windows.
3) Enter SE59XX_TARGET_IP via ssh using software putty.
4) Login account as first step shown.

2.9.2 Use a debug command line to Login

If you're not pressing "Esc" button within 3 seconds from boot-up, the device will enter Linux login mode as per screenshot below

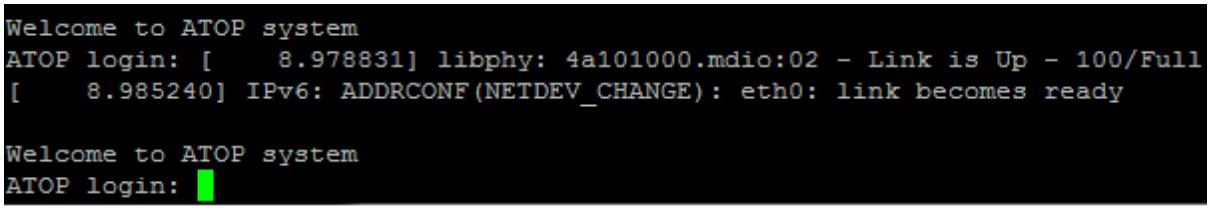


Figure 2.16 Command line Login

2.10 Automatic execution on Startup of Custom-Applications

1) Put your startup script "user_pre.sh or user_post.sh" into /jffs2 of root file system via FTP or TFTP.
2) SE5904D will execute both user_pre.sh and user_post.sh" after startup from next reboot.

2.11 Startup messages

The following is the standard startup message from SE5904D (as example):

```
U-Boot 2014.07-svn332 (Feb 15 2017 - 13:39:25)

I2C:   ready
DRAM:  512 MiB
Flash: 32 MiB
MMC:   OMAP SD/MMC: 0, OMAP SD/MMC: 1
*** Warning - bad CRC, using default environment

Net:   cpsw
Hit ESC to execute ATOP menu:
Wait ... 0
Booting from ramdisk ...
Kernel image @ 0x82000000 [ 0x000000 - 0x383458 ]
## Loading init Ramdisk from Legacy Image at 84080000 ...
   Image Name:   RootFS
   Created:      2017-02-15   5:38:49 UTC
   Image Type:   ARM Linux RAMDisk Image (gzip compressed)
   Data Size:    6787658 Bytes = 6.5 MiB
   Load Address: 00000000
   Entry Point:  00000000
   Verifying Checksum ... OK
## Flattened Device Tree blob at 84000000
   Booting using the fdt blob at 0x84000000
   Loading Ramdisk to 87986000, end 87fff24a ... OK
   Loading Device Tree to 8797a000, end 879850fe ... OK

Starting kernel ...

[    0.000000] Booting Linux on physical CPU 0x0
[    0.000000] Linux version 3.14.26-svn630 (willylin@ubuntu-test) (gcc version 4.7.3 20130226 (prerelease) (crosstool-NG linaro-
1.13.1-4.7-2013.03-20130313 - Linaro GCC 2013.03) ) #24 Wed Feb 15 13:35:45 CST 2017
[    0.000000] CPU: ARMv7 Processor [413fc082] revision 2 (ARMv7), cr=10c5387d
[    0.000000] CPU: PIPT / VIPT nonaliasing data cache, VIPT aliasing instruction cache
[    0.000000] Machine model: TI AM335x EVM
[    0.000000] cma: CMA: reserved 16 MiB at 9e800000
[    0.000000] Memory policy: Data cache writeback
[    0.000000] CPU: All CPU(s) started in SVC mode.
[    0.000000] AM335X ES2.1 (sgx neon )
[    0.000000] Built 1 zonelists in Zone order, mobility grouping on.  Total pages: 129792
[    0.000000] Kernel command line: console=ttyO0,115200n8 ramdisk_size=800000 root=/dev/ram0 rw bootloader_ver=1.0
rootfstype=ext2
[    0.000000] PID hash table entries: 2048 (order: 1, 8192 bytes)
[    0.000000] Dentry cache hash table entries: 65536 (order: 6, 262144 bytes)
[    0.000000] Inode-cache hash table entries: 32768 (order: 5, 131072 bytes)
[    0.000000] Memory: 487916K/523264K available (4921K kernel code, 271K rwdata, 1772K rodata, 278K init, 387K bss, 35348K
reserved, 0K highmem)
[    0.000000] Virtual kernel memory layout:
[    0.000000]     vector  : 0xffff0000 - 0xffff1000   (   4 kB)
[    0.000000]     fixmap  : 0xfff00000 - 0xfffe0000   ( 896 kB)
[    0.000000]     vmalloc : 0xe0800000 - 0xff000000   ( 488 MB)
[    0.000000]     lowmem  : 0xc0000000 - 0xe0000000   ( 512 MB)
[    0.000000]     pkmap   : 0xbfe00000 - 0xc0000000   (   2 MB)
[    0.000000]     modules : 0xbf000000 - 0xbfe00000   (  14 MB)
[    0.000000]       .text : 0xc0008000 - 0xc06916a4   (6694 kB)
[    0.000000]       .init : 0xc0692000 - 0xc06d7a0c   ( 279 kB)
[    0.000000]       .data : 0xc06d8000 - 0xc071be18   ( 272 kB)
[    0.000000]        .bss : 0xc071be18 - 0xc077cb00   ( 388 kB)
[    0.000000] NR_IRQS:16 nr_irqs:16 16
[    0.000000] IRQ: Found an INTC at 0xfa200000 (revision 5.0) with 128 interrupts
```

```
[    0.000000] Total of 128 interrupts on 1 active controller
[    0.000000] OMAP clockevent source: timer2 at 24000000 Hz
[    0.000013] sched_clock: 32 bits at 24MHz, resolution 41ns, wraps every 178956969942ns
[    0.000033] OMAP clocksource: timer1 at 24000000 Hz
[    0.000293] Console: colour dummy device 80x30
[    0.000318] Calibrating delay loop... 794.62 BogoMIPS (lpj=397312)
[    0.006470] pid_max: default: 32768 minimum: 301
[    0.006570] Security Framework initialized
[    0.006626] Mount-cache hash table entries: 1024 (order: 0, 4096 bytes)
[    0.006638] Mountpoint-cache hash table entries: 1024 (order: 0, 4096 bytes)
[    0.013453] CPU: Testing write buffer coherency: ok
[    0.013874] Setting up static identity map for 0x804a8630 - 0x804a86a0
[    0.015073] devtmpfs: initialized
[    0.016818] VFP support v0.3: implementor 41 architecture 3 part 30 variant c   rev 3
[    0.022954] omap_hwmod: tptc0 using broken dt data from edma
[    0.023029] omap_hwmod: tptc1 using broken dt data from edma
[    0.023093] omap_hwmod: tptc2 using broken dt data from edma
[    0.027050] omap_hwmod: debugss: _wait_target_disable failed
[    0.080891] pinctrl core: initialized pinctrl subsystem
[    0.082028] regulator-dummy: no parameters
[    0.083870] NET: Registered protocol family 16
[    0.085826] DMA: preallocated 256 KiB pool for atomic coherent allocations
[    0.087423] cpuidle: using governor ladder
[    0.087438] cpuidle: using governor menu
[    0.095936] platform 49000000.edma: alias fck already exists
[    0.095960] platform 49000000.edma: alias fck already exists
[    0.095972] platform 49000000.edma: alias fck already exists
[    0.097087] OMAP GPIO hardware version 0.1
[    0.106842] omap-gpmc 50000000.gpmc: could not find pctldev for node /pinmux@44e10800/norflash_pins_default, deferring
probe
[    0.106875] platform 50000000.gpmc: Driver omap-gpmc requests probe deferral
[    0.110401] No ATAGs?
[    0.110420] hw-breakpoint: debug architecture 0x4 unsupported.
[    0.127584] bio: create slab <bio-0> at 0
[    0.141124] edma-dma-engine edma-dma-engine.0: TI EDMA DMA engine driver
[    0.142127] lis3_reg: no parameters
[    0.143476] usbcore: registered new interface driver usbfs
[    0.143657] usbcore: registered new interface driver hub
[    0.143889] usbcore: registered new device driver usb
[    0.144782] omap_i2c 44e0b000.i2c: could not find pctldev for node /pinmux@44e10800/pinmux_i2c0_pins, deferring probe
[    0.144808] platform 44e0b000.i2c: Driver omap_i2c requests probe deferral
[    0.144834] omap_i2c 4802a000.i2c: could not find pctldev for node /pinmux@44
e10800/pinmux_i2c1_pins, deferring probe
[    0.144847] platform 4802a000.i2c: Driver omap_i2c requests probe deferral
[    0.146415] omap-mailbox 480c8000.mailbox: omap mailbox rev 0x400
[    0.148148] Switched to clocksource timer1
[    0.184253] NET: Registered protocol family 2
[    0.185043] TCP established hash table entries: 4096 (order: 2, 16384 bytes)
[    0.185101] TCP bind hash table entries: 4096 (order: 2, 16384 bytes)
[    0.185189] TCP: Hash tables configured (established 4096 bind 4096)
[    0.185270] TCP: reno registered
[    0.185283] UDP hash table entries: 256 (order: 0, 4096 bytes)
[    0.185301] UDP-Lite hash table entries: 256 (order: 0, 4096 bytes)
[    0.185496] NET: Registered protocol family 1
[    0.185811] RPC: Registered named UNIX socket transport module.
[    0.185822] RPC: Registered udp transport module.
[    0.185828] RPC: Registered tcp transport module.
[    0.185834] RPC: Registered tcp NFSv4.1 backchannel transport module.
[    0.186758] Trying to unpack rootfs image as initramfs...
[    0.187541] rootfs image is not initramfs (no cpio magic); looks like an initrd
[    0.231677] Freeing initrd memory: 6628K (c7986000 - c7fff000)
[    0.231976] hw perfevents: enabled with ARMv7 Cortex-A8 PMU driver, 5 counters available
[    0.235247] futex hash table entries: 256 (order: -1, 3072 bytes)
[    0.378226] NFS: Registering the id_resolver key type
[    0.378317] Key type id_resolver registered
[    0.378326] Key type id_legacy registered
```

```
[    0.378364] jffs2: version 2.2. (NAND) (SUMMARY)  © 2001-2006 Red Hat, Inc.
[    0.378550] msgmni has been set to 997
[    0.379395] io scheduler noop registered
[    0.379410] io scheduler deadline registered
[    0.379431] io scheduler cfq registered (default)
[    0.381146] pinctrl-single 44e10800.pinmux: 142 pins at pa f9e10800 size 568
[    0.382600] Serial: 8250/16550 driver, 48 ports, IRQ sharing enabled
[    0.400065] omap_uart 44e09000.serial: no wakeirq for uart0
[    0.400264] 44e09000.serial: ttyO0 at MMIO 0x44e09000 (irq = 88, base_baud = 3000000) is a OMAP UART0
[    1.028894] console [ttyO0] enabled
[    1.033403] omap_uart 48022000.serial: no wakeirq for uart0
[    1.039392] 48022000.serial: ttyO1 at MMIO 0x48022000 (irq = 89, base_baud = 3000000) is a OMAP UART1
[    1.050604] omap_rng 48310000.rng: OMAP Random Number Generator ver. 20
[    1.068218] brd: module loaded
[    1.076954] loop: module loaded
[    1.080664] (hci_tty): inside hci_tty_init
[    1.085470] (hci_tty): allocated 251, 0
[    1.091197] Atop update device initialize success.
[    1.096857] 2000020.atop_relay:
[    1.100095] Register atop_relay0 success.
[    1.105516] 200000e.atop_cpld:
[    1.108659] Register atop_cpld_ver0 success.
[    1.115365] mtdoops: mtd device (mtddev=name/number) must be supplied
[    1.125434] usbcore: registered new interface driver cdc_ether
[    1.131779] usbcore: registered new interface driver smsc95xx
[    1.137863] ehci_hcd: USB 2.0 'Enhanced' Host Controller (EHCI) Driver
[    1.144735] ehci-omap: OMAP-EHCI Host Controller driver
[    1.150487] ohci_hcd: USB 1.1 'Open' Host Controller (OHCI) Driver
[    1.157004] ohci-platform: OHCI generic platform driver
[    1.164668] 47401300.usb-phy supply vcc not found, using dummy regulator
[    1.174193] musb-hdrc musb-hdrc.0.auto: Failed to request rx1.
[    1.180400] musb-hdrc musb-hdrc.0.auto: musb_init_controller failed with status -517
[    1.188587] platform musb-hdrc.0.auto: Driver musb-hdrc requests probe deferral
[    1.196932] 47401b00.usb-phy supply vcc not found, using dummy regulator
[    1.206284] musb-hdrc musb-hdrc.1.auto: Failed to request rx1.
[    1.212478] musb-hdrc musb-hdrc.1.auto: musb_init_controller failed with status -517
[    1.220652] platform musb-hdrc.1.auto: Driver musb-hdrc requests probe deferral
[    1.242648] input: beeper.7 as /devices/beeper.7/input/input0
[    1.250121] omap_rtc 44e3e000.rtc: rtc core: registered 44e3e000.rtc as rtc0
[    1.258078] i2c /dev entries driver
[    1.264330] oprofile: using arm/armv7
[    1.268471] TCP: cubic registered
[    1.271973] Initializing XFRM netlink socket
[    1.276527] NET: Registered protocol family 10
[    1.282075] sit: IPv6 over IPv4 tunneling driver
[    1.289180] NET: Registered protocol family 17
[    1.293908] NET: Registered protocol family 15
[    1.298623] 8021q: 802.1Q VLAN Support v1.8
[    1.303099] Key type dns_resolver registered
[    1.308461] ondemand governor failed, too long transition latency of HW, fallback to performance governor
[    1.319330] PM: bootloader does not support rtc-only!
[    1.324662] ThumbEE CPU extension supported.
[    1.329187] Registering SWP/SWPB emulation handler
[    1.335887] lis3_reg: disabling
[    1.339233] regulator-dummy: disabling
[    1.343584] omap-gpmc 50000000.gpmc: GPMC revision 6.0
[    1.349016] gpmc_mem_init: disabling cs 0 mapped at 0x0-0x1000000
[    1.357218] spansion,s29gl256p11t: Found 1 x16 devices at 0x0 in 16-bit bank. Manufacturer ID 0x000001 Chip ID 0x002201
[    1.368582] Amd/Fujitsu Extended Query Table at 0x0040
[    1.374002]   Amd/Fujitsu Extended Query version 1.3.
[    1.379317]   Advanced Sector Protection (PPB Locking) supported
[    1.385627] number of CFI chips: 1
[    1.389506] 6 ofpart partitions found on MTD device spansion,s29gl256p11t
[    1.396663] Creating 6 MTD partitions on "spansion,s29gl256p11t":
[    1.403078] 0x000000000000-0x000000080000 : "u-boot"
[    1.409937] 0x000000080000-0x0000000a0000 : "u-boot env"
```

```
[    1.416983] 0x0000000a0000-0x000000100000 : "DTB"
[    1.423378] 0x000000100000-0x000000600000 : "kernel"
[    1.430032] 0x000000600000-0x000001000000 : "rootfs"
[    1.436704] 0x000001000000-0x000001fe0000 : "jffs2"
[    1.446077] at24 0-0050: 8192 byte 24c64 EEPROM, writable, 1 bytes/write
[    1.646744] rtc-pcf8563 0-0051: chip found, driver version 0.4.3
[    1.653514] rtc-pcf8563 0-0051: low voltage detected, date/time is not reliable.
[    1.661674] rtc-pcf8563 0-0051: rtc core: registered rtc-pcf8563 as rtc1
[    1.668758] omap_i2c 44e0b000.i2c: bus 0 rev0.11 at 400 kHz
[    1.675743] omap_i2c 4802a000.i2c: bus 1 rev0.11 at 100 kHz
[    1.685059] musb-hdrc musb-hdrc.0.auto: MUSB HDRC host driver
[    1.691135] musb-hdrc musb-hdrc.0.auto: new USB bus registered, assigned bus number 1
[    1.699605] usb usb1: New USB device found, idVendor=1d6b, idProduct=0002
[    1.706757] usb usb1: New USB device strings: Mfr=3, Product=2, SerialNumber= 1
[    1.714348] usb usb1: Product: MUSB HDRC host driver
[    1.719566] usb usb1: Manufacturer: Linux 3.14.26-svn630 musb-hcd
[    1.725969] usb usb1: SerialNumber: musb-hdrc.0.auto
[    1.732459] hub 1-0:1.0: USB hub found
[    1.736472] hub 1-0:1.0: 1 port detected
[    1.744435] musb-hdrc musb-hdrc.1.auto: MUSB HDRC host driver
[    1.750505] musb-hdrc musb-hdrc.1.auto: new USB bus registered, assigned bus  number 2
[    1.758919] usb usb2: New USB device found, idVendor=1d6b, idProduct=0002
[    1.766080] usb usb2: New USB device strings: Mfr=3, Product=2, SerialNumber=1
[    1.773674] usb usb2: Product: MUSB HDRC host driver
[    1.778893] usb usb2: Manufacturer: Linux 3.14.26-svn630 musb-hcd
[    1.785287] usb usb2: SerialNumber: musb-hdrc.1.auto
[    1.791585] hub 2-0:1.0: USB hub found
[    1.795605] hub 2-0:1.0: 1 port detected
[    1.847188] davinci_mdio 4a101000.mdio: davinci mdio revision 1.6
[    1.853618] davinci_mdio 4a101000.mdio: detected phy mask fffffff9
[    1.861456] libphy: 4a101000.mdio: probed
[    1.865701] davinci_mdio 4a101000.mdio: phy[1]: device 4a101000.mdio:01, driver unknown
[    1.874105] davinci_mdio 4a101000.mdio: phy[2]: device 4a101000.mdio:02, driver unknown
[    1.883214] cpsw 4a100000.ethernet: Missing dual_emac_res_vlan in DT.
[    1.890000] cpsw 4a100000.ethernet: Using 1 as Reserved VLAN for 0 slave
[    1.897074] cpsw 4a100000.ethernet: Missing dual_emac_res_vlan in DT.
[    1.903867] cpsw 4a100000.ethernet: Using 2 as Reserved VLAN for 1 slave
[    1.910917] cpsw 4a100000.ethernet: Detected MACID = 50:65:83:57:8a:4f
[    1.919759] cpsw 4a100000.ethernet: cpsw: Detected MACID = 50:65:83:57:8a:51
[    1.929574] input: gpio_keys.8 as /devices/gpio_keys.8/input/input1
[    1.937356] rtc-pcf8563 0-0051: low voltage detected, date/time is not reliable.
[    1.945176] rtc-pcf8563 0-0051: setting system clock to 2017-01-03 21:41:31 UTC (1483479691)
[    1.957722] RAMDISK: gzip image found at block 0
[    2.121316] usb 1-1: new full-speed USB device number 2 using musb-hdrc
[    2.247211] usb 1-1: New USB device found, idVendor=04e2, idProduct=1414
[    2.254276] usb 1-1: New USB device strings: Mfr=0, Product=0, SerialNumber=0
[    2.858767] VFS: Mounted root (ext2 filesystem) on device 1:0.
[    2.865213] devtmpfs: mounted
[    2.868733] Freeing unused kernel memory: 276K (c0692000 - c06d7000)
Starting logging: OK
Initializing random number generator... [    2.998416] random: dd urandom read with 23 bits of entropy available
done.
[    3.186318] jffs2: notice: (781) jffs2_build_xattr_subsystem: complete building xattr subsystem, 0 of xdatum (0 unchecked, 0
orphan) and 0 of xref (0 dead, 0 orphan) found.
logger: unknown facility name: local
[    3.234911] cdc_xr_usb_serial 1-1:1.0: This device cannot do calls on its own . It is not a modem.
[    3.245231] cdc_xr_usb_serial 1-1:1.0: ttyXR_USB_SERIAL0: USB XR_USB_SERIAL device
[    3.255101] cdc_xr_usb_serial 1-1:1.2: This device cannot do calls on its own. It is not a modem.
[    3.265200] cdc_xr_usb_serial 1-1:1.2: ttyXR_USB_SERIAL1: USB XR_USB_SERIAL device
[    3.274703] cdc_xr_usb_serial 1-1:1.4: This device cannot do calls on its own. It is not a modem.
[    3.284737] cdc_xr_usb_serial 1-1:1.4: ttyXR_USB_SERIAL2: USB XR_USB_SERIAL device
[    3.294217] cdc_xr_usb_serial 1-1:1.6: This device cannot do calls on its own. It is not a modem.
[    3.305684] cdc_xr_usb_serial 1-1:1.6: ttyXR_USB_SERIAL3: USB XR_USB_SERIAL device
[    3.317579] usbcore: registered new interface driver cdc_xr_usb_serial
[    3.324481] xr: Exar USB UART (serial port) driver
Starting Monit 5.18 daemon
```

```
Starting network...
[    6.877040] device eth0 entered promiscuous mode
[    6.909977] net eth0: initializing cpsw version 1.12 (0)
[    6.971560] net eth0: phy found : id is : 0x2000a212
[    6.979199] IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready
[    6.985310] 8021q: adding VLAN 0 to HW filter on device eth0
net.ipv6.conf.eth0.accept_ra = 1
[    7.048551] device eth1 entered promiscuous mode
[    7.090144] net eth1: initializing cpsw version 1.12 (0)
[    7.150504] net eth1: phy found : id is : 0x2000a212
[    7.156080] IPv6: ADDRCONF(NETDEV_UP): eth1: link is not ready
[    7.162240] 8021q: adding VLAN 0 to HW filter on device eth1
net.ipv6.conf.eth1.accept_ra = 1
Starting atop_ntp.sh: OK
Starting dropbear sshd: OK
Starting ProFTPD: [    7.332967] warning: `proftpd' uses 32-bit capabilities (legacy support in use)
done
Starting stunnel: [.] stunnel 5.09 on arm-buildroot-linux-gnueabihf platform
[.] Compiled/running with OpenSSL 1.0.2 22 Jan 2015
[.] Threading:FORK Sockets:POLL,IPv6 TLS:ENGINE,FIPS,OCSP,PSK,SNI
[ ] errno: (*__errno_location ())
[.] Reading configuration from file /etc/stunnel/stunnel.conf
[.] UTF-8 byte order mark detected
[ ] Enabling support for engine "capi"
[!] error queue: 2606A074: error:2606A074:engine routines:ENGINE_by_id:no such engine
[!] ENGINE_by_id: 260B606D: error:260B606D:engine routines:DYNAMIC_LOAD:init failed
[!] Line 18: "engine = capi": Failed to open the engine
FAIL
Starting network management services: snmpd.

Welcome to ATOP system
ATOP login:
```

3 Hardware Specifications

3.1 Packing List

Inside the purchased package, you will find the following items:

Table 3.1 Packing List

Item	Quantity	Description
SE59XX	1	Industrial Serial Device Server
Mounting Kit	1	On SE5908 / SE5916 / SE5908A / SE5916A Rack Mounting Type-L angles)x 2(Screws)x 6(On SE5901 / SE5904D / SE5901B - DIN Rail Kit
Terminal Block		Power Supply/ Relay output: TB3 x 1: 3-pin 5.08mm lockable Terminal Block (SE5901, SE5901B) TB3 x 2: 3-pin 5.08mm lockable Terminal Block (SE5908-DC,SE5916-DC) TB7 x1: 7-pin 5.08mm lockable Terminal Block (SE5904D only) Serial ports: Terminal block is included only on TB model TB5 x 1: 5-pin 5.08mm lockable Terminal Block (SE5901) TB5 x 4: 5-pin 5.08mm lockable Terminal Block (SE5904D) TB5 x 8: 5-pin 5.08mm lockable Terminal Block (SE5908A) TB5 x 16: 5-pin 5.08mm lockable Terminal Block (SE5916A)
Documentation	1	Hardware Installation Guide)Warranty card is included(
Mounting Kit	1	DIN-Rail Kit (Already mounted on the device)

Note: Please notify your sales representative if any of the above items is missing or damaged in any form upon delivery. If your sales representative is unable to satisfy your enquiries, please contact us directly.

3.2 Optional Accessories

The following table lists optional accessories for SE59XX SDK series.

Table 3.2 Optional Accessories

Item	Description
UN315-1212(US-LDC)	Y-Type (5.08mm) power adapter, 100-240VAC input, 1.25A @ 12VDC output, US plug
UNE315-1212(EU-LDC)	Y-Type (5.08mm) power adapter, 100-240VAC input, 1.25A @ 12VDC output, EU plug
ADP-DB9(F)-TB5	Female DB9 to Female 3.81 TB5 Converter
CBL-RJ45(8P)-DB9(F)	8-pin RJ45-DB9 debug cable, 90cm
GDC-120	120mm copper woven grounding cable

LM28-C3S-TI-N	SFP Transceiver, 1250Mbps, 850nmVCSEL, Multi-mode, 550m, 3.3V, -20~85°C
LM38-C3S-TI-N	SFP Transceiver, 1250Mbps, 1310nmFP, Multi-mode, 2km, 3.3V, -40~85°C
LS38-C3S-TI-N	SFP Transceiver, 1250Mbps, 1310nmFP, Single-mode, 10km, 3.3V, -40~85°C
LS38-C3L-TI-N	SFP Transceiver, 1250Mbps, 1310nmDFB, Single-mode, 30km, 3.3V, -40~85°C
WMK-450-Black	Black Aluminum Wall Mount Kit (DIN-rail items only)

3.3 Hardware

Table 3.3 Hardware features

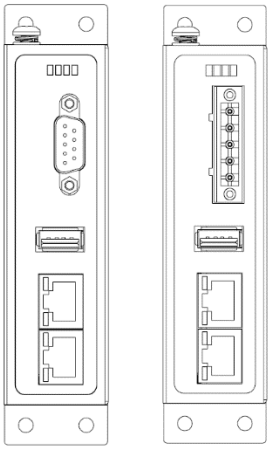
System	
CPU	32-bit ARM Based TI CPU AM3354 800MHz (except SE5908A/SE5916A use AM3352 1GHz)
Flash Memory	32MB
RAM	SE5901 DDR2 128MB SE5901B DDR2 256MB SE5904D DDR3 256MB SE5900A/08A/16A/MB5908/16 DDR3 256MB
EEPROM	8 KB
Reset	Built-in Recessed Key (Restore to Factory Defaults)
Watchdog	Hardware built-in
Network	
Ethernet Interface	IEEE 802.3 10BaseT IEEE 802.3u 100BaseT(X) IEEE 802.3ac 1000BaseT(X) – SFP version of SE5904D only IEEE 802.3af (PoE PD) –selected SE5901 and SE5904D versions can be powered through PoE Connection: SFP or RJ45
Serial	
Serial Interface	RS-232/RS-422/RS-485 Software Selectable (Default: RS-232) • The first port available on SE5901B is RS-232/RS-485 • The second port available on SE5901B-IO-X is only RS-232 • The isolation version (-SiS) on SE5908/SE5916/SE5908A/ SE5916A supports only RS-422/ RS-485
Serial Connector	Connector Type • SE5916 -16 Serial Ports (RJ45) • SE5908 - 8 Serial Ports (RJ45) • SE5916A – 16 Serial Ports (TB-5 or DB-9) • SE5908A – 8 Serial Ports (TB-5 or DB-9) • SE5904 – 4 Serial Ports (TB-5 or DB-9) • SE5901 – 1 Serial Port (TB-5 or DB-9) • SE5901B – 1 Serial Port (TB-14 or DB-9) – includes I/O
Protection	SE5901/SE5901B no isolation SE5904D/ SE5908A/16A (optional 3V) SE5908/16 (optional 2.5kV)
Serial Port Communication	Baud-rate: 1200 bps ~ 921600 bps Parity: None, Even, Odd, Mark, or Space Data Bits: 5, 6, 7, 8 Stop Bits: 1, 2 Software Selectable Flow Control: RTS/CTS (RS-232 only), XON/XOFF, None

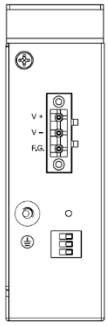
LED Indicator		
LED indication	Power x 2 (SE5901- SE5901B – SE5908 – SE5916 x 1) RUN x 1 ALARM x 1 LAN: • x 2 (all versions except SE5908A and SE5916A) • x 6 (SE5908A and SE5916A only) COM port: • x 16 (SE5916 and SE5916A); • x 8 (SE5908 and SE5908A); • x 4 (SE5904D); x 1 (SE5901 and SE5901B)	
Power Requirement & EMC		
Input	SE5908/ SE5916 : • Single 100~240 VAC (EU/US versions) • Single 24~48 VDC (DC version) SE5908A/ SE5916A • Redundant 100~240 VAC or 100~370 VDC (TB)– HV vers. • Redundant 24~48 VDC- DC version SE5901/SE5901B : Single 9~48 VDC SE5904D : Redundant 9~48 VDC	
Consumption	Max.17.5 W (SE5908 /SE5916) Max. 6W (SE5901) Max. 7.8W(SE5904D) Max. 17.5W(SE5908A/SE5916A) Max. 7.2W(SE5901B)	
EMI/EMC	FCC Part 15, Subpart B, Class A EN 55032, Class B, EN 61000-6-2, Class B EN 61000-3-2, EN 61000-3-3 EN 55024, EN 61000-6-4 IEC 61850-3 / IEEE 1613 (SE5908A and SE5916A only)	
Mechanical		
Dimensions (W x H x D, mm)	SE5901: 32 mm x 110 mm x 90 mm (1.26 x 4.33 x 3.54 in) SE5901B: 32 mm x 122mm x 92 mm (1.26 x 4.8 x 3.62 in) SE5904D: 55 mm x 145 mm x 113mm (2.17 x 5.17 x 4.45 in) SE5908: 436 mm x 43.5 mm x 200 mm (17.17 x 1.71 x 7.87 in) SE5916: 436 mm x 43.5 mm x 200 mm (17.17 x 1.71 x 7.87 in) SE5908A: 440.6mm x 44 mm x 309 mm (17.35 x 1.73 x 12.17 in) SE5916A: 440.6mm x 44 mm x 309 mm (17.35 x 1.73 x 12.17 in)	
Enclosure	IP30 protection, metal housing	
Environmental		
Temperature	Operations	-40°C ~ 85°C (-40°F ~ 185°F) (except SE5901B -40°C ~ 70°C and SE5908/SE5916 -20°C ~ 70°C)
	Storage	-40°C ~ 85°C (-40°F ~ 185°F)
Relative Humidity	5% ~ 95%, 55°C Non-condensing	

3.4 External Device's Overview

The following figures show particular SE59XX series device's front and rear panels.

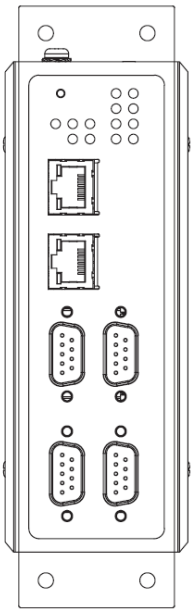
SE5901



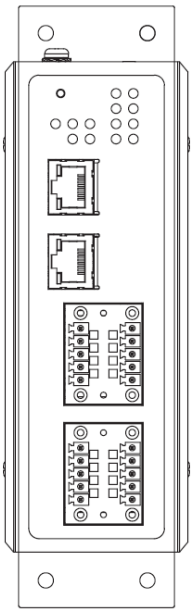


SE5904D

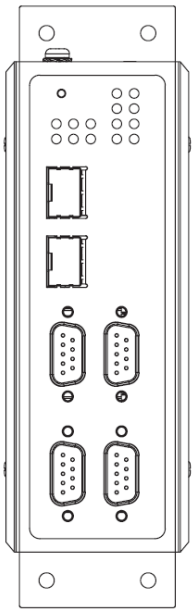
RJ45 DB9 Type



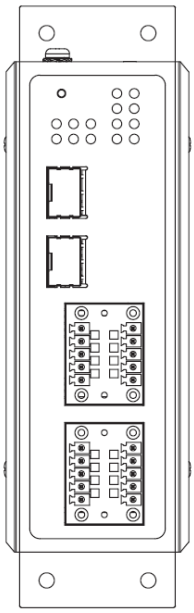
RJ45 TB5 Type

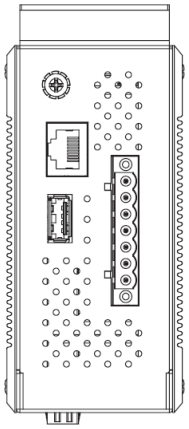


SFP DB9 Type

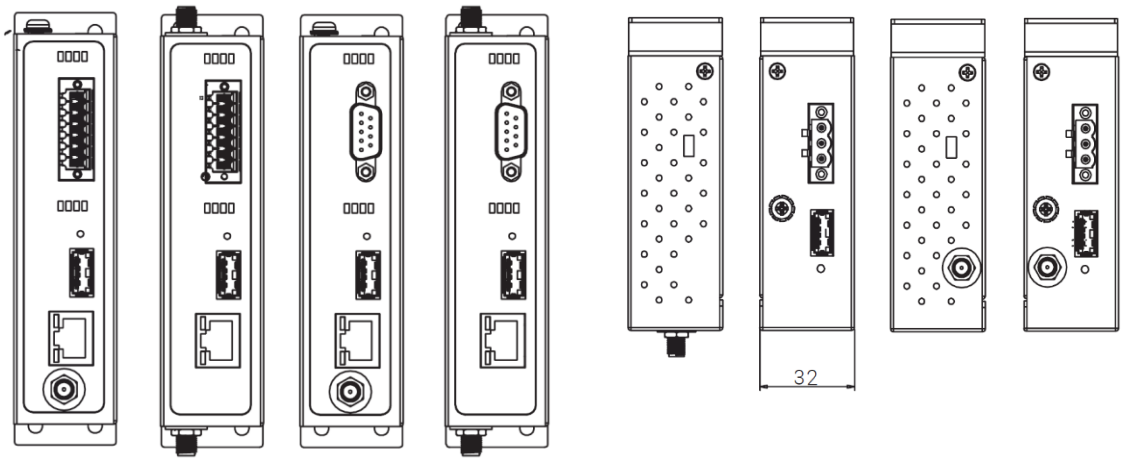


SFP TB5 Type

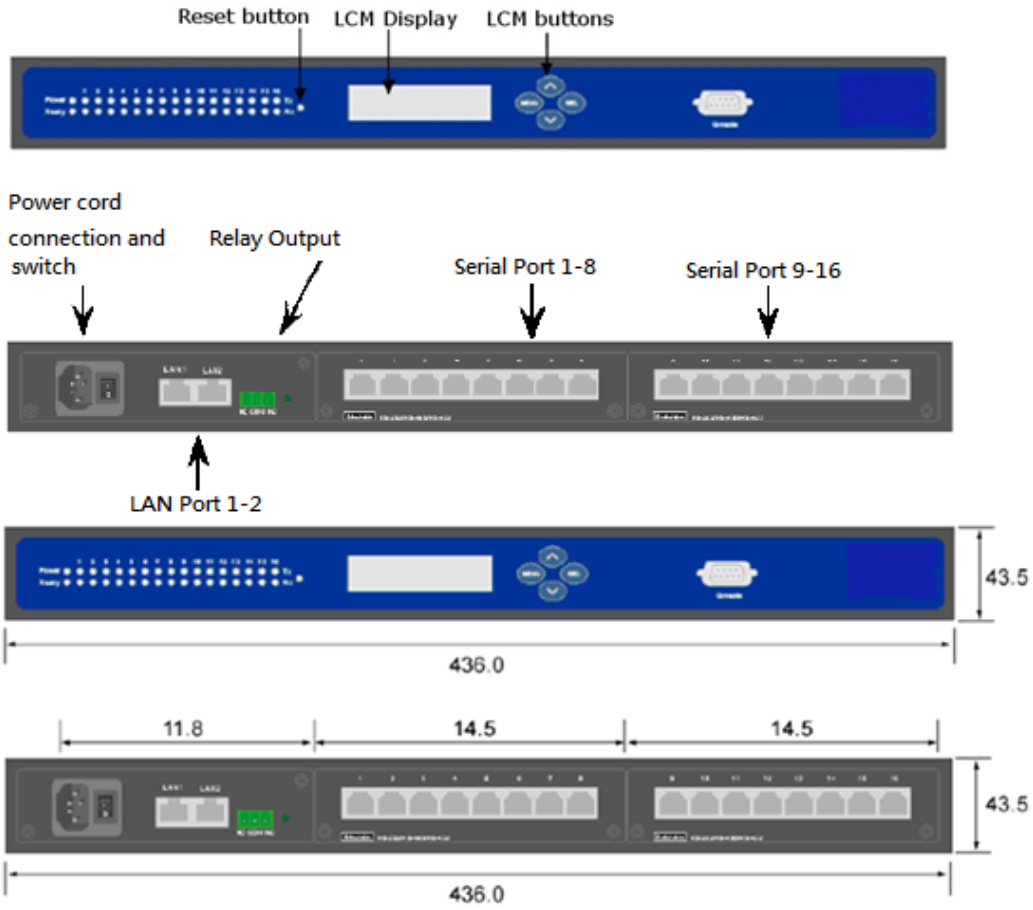




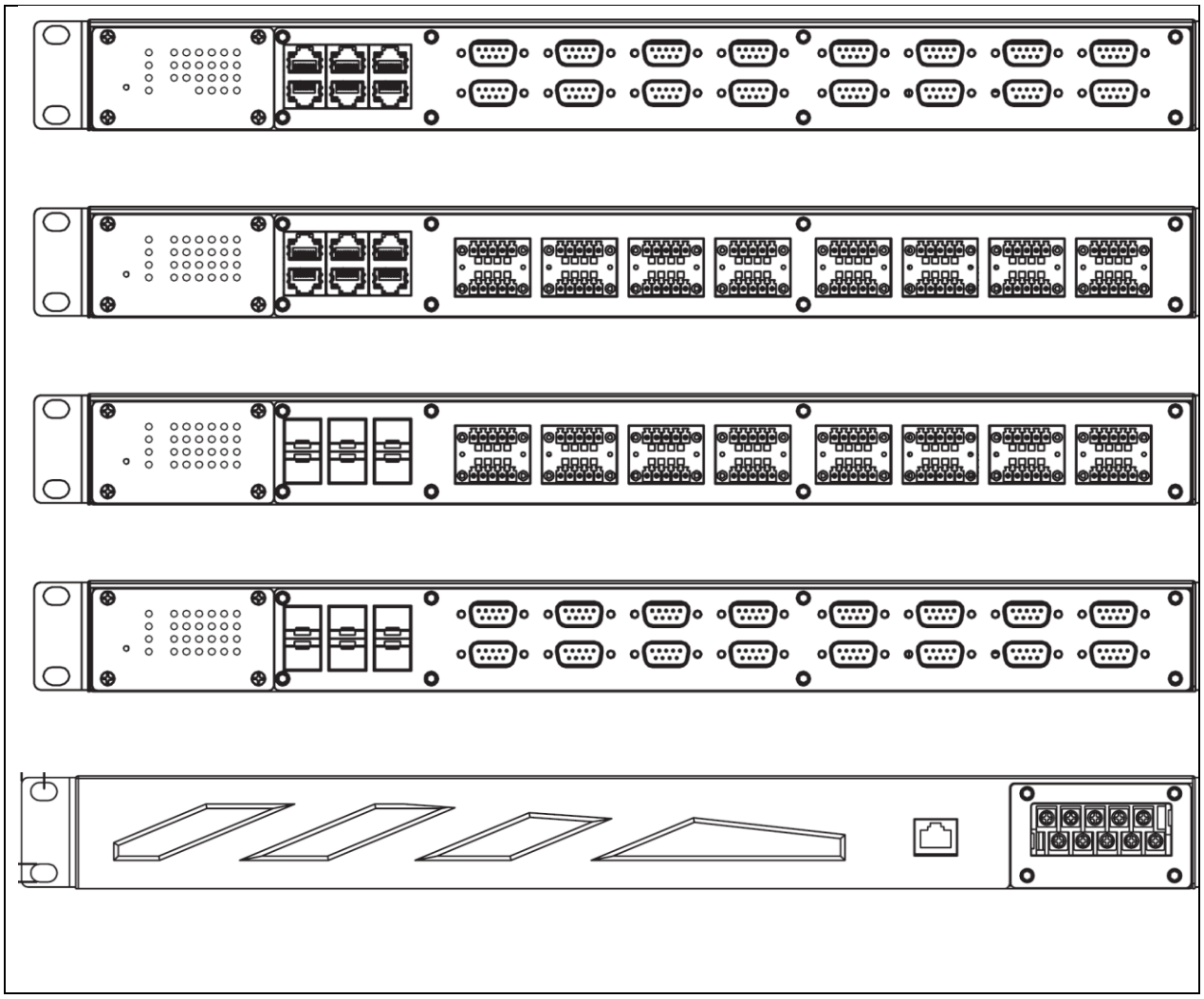
SE5901B



SE5908/16


Reset button
LCM Display
LCM buttons
Power cord
connection and
switch
Relay Output
Serial Port 1-8
Serial Port 9-16
LAN Port 1-2
43.5
436.0
11.8
14.5
14.5
43.5
436.0

SE5908A/16A



3.5 Serial Pin Assignments

3.5.1 SE5901 Pin Assignments for Serial Interfaces

DB9 to RS-232/RS-422/RS-485 connectors

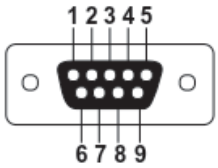


Figure 3.1 DB9 Pin Number

Table 3.4 SE5901 Pin Assignment for DB9 to RS-232/RS-422/RS-485 Connector

Pin#	RS-232 Full Duplex	RS-422/4-Wire RS-485 Full Duplex	2-Wire RS-485 Half Duplex
1	DCD	N/A	N/A
2	RxD	TXD+	N/A
3	TxD	RXD+	Data+
4	DTR	N/A	N/A
5	SG (Signal Ground)	SG (Signal Ground)	SG (Signal Ground)
6	DSR	N/A	N/A
7	RTS	RXD-	Data-
8	CTS	TXD-	N/A
9	RI	N/A	N/A

1 x 5-pin (Male Terminal Block) for RS-232/RS-422/RS485 Connector

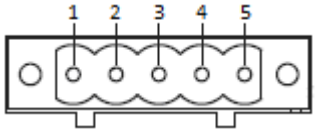


Figure 3.2 TB5 Pin Number

Table 3.5 SE5901 Pin Assignment for TB5 to RS-232/RS-422/RS-485 Connector

Pin#	RS-232 Full Duplex	RS-422/4-Wire RS-485 Full Duplex	2-Wire RS-485 Half Duplex
1	RxD	T+	NC
2	CTS	T-	NC
3	TxD	R+	Data+
4	RTS	R-	Data-
5	SG (Signal Ground)	SG (Signal Ground)	SG (Signal Ground)

3.5.2 SE5904D Pin Assignments

DB9 to RS-232/RS-485/RS-422 connectors

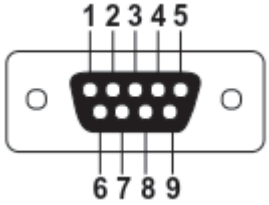


Figure 3.3 DB9 Pin Number

Table 3.6 MB5904D Pin Assignment for DB9 to RS-232/RS422/RS-485 Connectors

Pin#	RS-232 Full Duplex	RS-422 Full Duplex	RS-485 Half Duplex
1	DCD	N/A	N/A
2	RxD	TxD+	Data+
3	TxD	RxD+	N/A
4	DTR	N/A	N/A
5	SG (Signal Ground)	SG (Signal Ground)	SG (Signal Ground)
6	DSR	N/A	N/A
7	RTS	RxD-	N/A
8	CTS	TxD-	Data-
9	RI	N/A	N/A

5-Pin Terminal Block to RS-485/RS-422 connectors

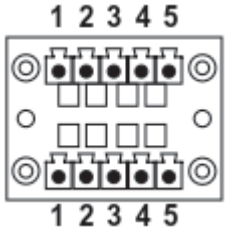


Figure 3.4 Terminal Block (TB-5) Pin Number

Table 3.7 MB5904D Pin Assignment for 5-Pin Terminal Block to RS-232/RS-422/RS-485 Connectors

Pin#	RS-232	RS-422 4-Wire RS-485	2-W RS-485
1	RxD	TxD+	Data+
2	CTS	TxD-	Data-

3	TxD	RxD+	N/A
4	RTS	RxD-	N/A
5	SG (Signal Ground)	SG (Signal Ground)	SG (Signal Ground)

3.5.3 SE5901B Pin Assignments

DB9 to RS-232/RS-485/RS-422 connectors

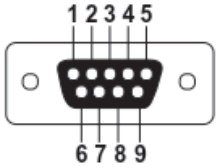


Figure 3.5 DB9 Pin Number

Table 3.8 SE5901B Pin Assignment for DB9 to RS-232/RS-485 Connector

Pin#	RS-232 Full Duplex	RS-485 Half Duplex
1	DCD	N/A
2	RxD	N/A
3	TxD	Data+
4	DTR	N/A
5	SG (Signal Ground)	SG (Signal Ground)
6	DSR	N/A
7	RTS	Data-
8	CTS	N/A
9	RI	N/A

2 x 7-pin Male Terminal Block for RS-232/485(COM 1),RS-232(COM 2) Relay and DI

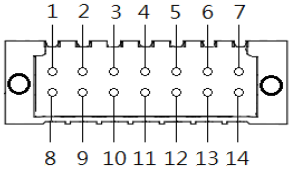


Figure 3.6 2 x 7-pin Male Terminal Block

Table 3.9 SE5901B 2 x 7-pin Male TB for RS-232/485(COM 1),RS-232(COM 2) Relay and DI pin-assignment

Pin#	DI and Relay	COM1 (RS-232)	COM1 (RS-485)	COM2 (RS-232)
1	DI1	*Dedicated for DI/DO*	*Dedicated for DI/DO*	*Dedicated for DI/DO*
2	DI2	*Dedicated for DI/DO*	*Dedicated for DI/DO*	*Dedicated for DI/DO*
3	Relay 1 -	*Dedicated for DI/DO*	*Dedicated for DI/DO*	*Dedicated for DI/DO*
4	Relay 1+	*Dedicated for DI/DO*	*Dedicated for DI/DO*	*Dedicated for DI/DO*
5	Relay 2 -	*Dedicated for DI/DO*	*Dedicated for DI/DO*	*Dedicated for DI/DO*
6	Relay 2+	*Dedicated for DI/DO*	*Dedicated for DI/DO*	*Dedicated for DI/DO*
7	*Dedicated for COM*	SG (Signal Ground)	SG (Signal Ground)	-
8	*Dedicated for COM*	Rx	-	-
9	*Dedicated for COM*	CTS	-	-
10	*Dedicated for COM*	Tx	Data +	-
11	*Dedicated for COM*	RTS	Data -	-
12	*Dedicated for COM*	-	-	SG (Signal Ground)

13	*Dedicated for COM*	-	-	Rx
14	*Dedicated for COM*	-	-	Tx

3.5.4 SE5908A/ SE5916A Pin Assignments

DB9 to RS-232/RS-485/RS-422 connectors

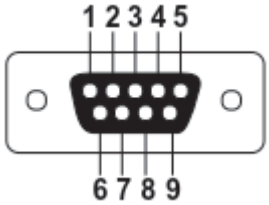


Figure 3.7 DB9 Pin Number

Table 3.10 SE5908A/16A Pin Assignment for DB9 to RS-232/RS422/RS-485 Connectors

Pin#	RS-232	RS-422	RS-485
1	-	-	-
2	RxD	TxD+	Data+
3	TxD	RxD+	-
4	-	-	-
5	SG (Signal Ground)	SG (Signal Ground)	SG (Signal Ground)
6	-	-	-
7	RTS	RxD-	-
8	CTS	TxD-	Data-
9	-	-	-

5-Pin Terminal Block to RS-232/RS-485/RS-422 connectors

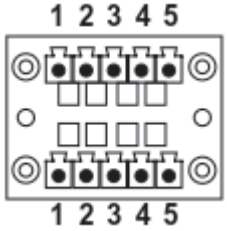


Figure 3.8 Terminal Block (TB-5) Pin Number

Table 3.11 SE5908A/16A Pin Assignment for 5-Pin Terminal Block to RS-232/RS-422/RS-485 Connectors

Pin#	RS-232	RS-422 4-Wire RS-485	2-W RS-485
1	RxD	TxD+	Data +
2	CTS	TxD-	Data -
3	TxD	RxD+	-
4	RTS	RxD-	-

5	SG (Signal Ground)	SG (Signal Ground)	SG (Signal Ground)

4 Software Specifications

The device node is the communication interface between user space and hardware device in Linux.
Each chapter is divided into two parts:

1. **How to program these interfaces** – The main purpose is to provide the way to access device node with some sample code.
2. **How to test the interface** – The main purpose is to describe the way to use Linux internal or Atop supporting tools to test the interfaces.

4.1 COM Port Interface

SE59XX Series (Except SE5900A) are equipped with COM ports. Each COM port is registered as a TTY terminal interface with the kernel.

- Maximum baud rate: 921600
- Minimum baud rate: 300
- Serial interface supported: RS232/RS485/RS422

The sample program is in the software/atop_application/utils/atop_loopback folder:

Table 4.1 Sample programs for COM port interface

File Name	Description
rs232_loopback.c	Loopback test program for RS232 ports
rs422_loopback.c	Loopback test program for RS422 ports
rs485_loopback.c	Loopback test program for RS485 ports

4.1.1 Program COM port interface

The following tables list the device node of COM port for each EVM model.

Table 4.2 SE59XX device node

Device node	Major & Minor number	Device Type	Description
ttyATOP0	266 0	Character	ATOP COM port 1
ttyATOP1	266 1	Character	ATOP COM port 2
ttyATOP2	266 2	Character	ATOP COM port 3
ttyATOP3	266 3	Character	ATOP COM port 4
…	…	…	…

Table 4.3 SE59XX Programming commands per device node

Device node	Ioctl command	Command Description
ttyATOP0-3	0x9000	Configure SE59XX COM port as one of RS232 / RS485 / RS422

Table 4.4 SE59XX ioctl command of COM Port

Ioctl command	parameter type	Value	Description
0x9000	integer	1	Configure to RS232 mode
		2	Configure to RS422 mode
		3	Configure to RS485 mode

In Linux system, user can use POSIX standard terminal interface to configure baud rate, data length, etc. It is called Termios and defined in system document <termios.h>. Please refer related Linux system document to configure it.

4.2 Network Interface

SE59XX Series are equipped with Network ports. The sample program in the folder software/atop_application/utils/atop_tcpserver describes how to use COM ports in combination with TCP server connections:

Table 4.5 Sample programs for TCP server connection to COM port communication

File Name	Description
atop_tcp_server.c	A sample program to use TCP server connection and COM port to make data communication.

4.3 Other Interfaces

There are multiple other interfaces available on SE59XX platform, depending on the actual hardware in use. Some devices are equipped with 4G connectivity, others with Relays and Digital inputs and soon. ATOP's convenient Software Development Kit is standardized for the whole family. We put at disposal simple programs that you can easily copy or emulate to make the best use of all interfaces.

All sample programs are in /atop_application/utils/atop_sdk folder:

4.3.1 Buzzer

There is one Buzzer in each SE59XX device. The sample program is available in the software/atop_application/utils/atop_sdk folder:

Table 4.6 Sample program for Buzzer

File Name	Description
buzzer.c	A sample program to use the device's Buzzer.

4.3.2 Digital Inputs

There are 2 Digital inputs on SE5901B-IO. The sample program is available in the software/atop_application/utils/atop_sdk folder:

Table 4.7 Sample program for Digital Input

File Name	Description
di_test.c	A sample program to use the device's Digital Inputs.

4.3.3 Digital Outputs

There are 2 Digital Outputs on SE5901B-IO. The sample program is available in the software/atop_application/utils/atop_sdk folder:

Table 4.8 Sample program for Digital Output

File Name	Description
do_test.c	A sample program to use the device's Digital Outputs.

4.3.4 Relay Outputs

There are Relay outputs on SE5904D, SE5908, SE5916, SE5900A, SE5908A and SE5916A. The sample program is available in the software/atop_application/utils/atop_sdk folder:

Table 4.9 Sample program for Relay Output

File Name	Description
relay.c	A sample program to use the device's Relay Outputs.

4.3.5 LCM (SE5908 / SE5916 only)

There is an LCM in SE5908 and SE5916. The sample program is available in the

software/atop_application/utils/atop_sdk folder:

Table 4.10 Sample program for LCM

File Name	Description
lcm_test.c	A sample program to use the device's LCM.

4.3.6 Reset Button

All SE59XX hardware platforms have a reset button. The sample program is available in the software/atop_application/utils/atop_sdk folder:

Table 4.11 Sample program for Reset Button

File Name	Description
button.c	A sample program to use the device's reset button.

4.3.7 Hardware Watchdog Timer

There is a hardware watchdog IC on each CPU board. If this IC is not reset within 1.6 seconds, then the system will reboot. This implementation allows the hardware to autonomously understand if the system is crashing, for whatever reason. During a System crash, the OS won't reset the IC within the deadline and therefore the system will automatically reboot.. All SE59XX hardware platforms do have an integrated hardware watchdog timer. The sample program is available in the software/atop_application/utils/atop_sdk folder:

Table 4.12 Sample program for WDT

File Name	Description
hwd.c	A sample program to use the device's Hardware Watchdog timer.

4.3.8 LEDs

Different devices in SE59XX family have different LEDs based on the number of ports. But all devices are equipped with a RUN/Fault LED. The sample program is available in the software/atop_application/utils/atop_sdk folder:

Table 4.13 Sample program for LEDs

File Name	Description
alarmLed.c	A sample program to use the device's Alarm (RED) LED
runLed.c	A sample program to use the device's Run (GREEN) LED

4.3.9 3G/4G Cellular (SE5901B only)

The sample program is available in the software/atop_application/utils/atop_sdk folder:

Table 4.14 Sample program for Cellular functions

File Name	Description
atop_4G_apn.c	A sample program to set the cellular Access Point
atop_4G_connect.c	A sample program to connect to 3G/4G
atop_4G_DialOnBoot.c	A sample program to set the device to dial on boot
atop_4G_PinDisable.c	A sample program to disable the SIM PIN
atop_4G_PinEnable.c	A sample program to enable the SIM PIN
atop_4G_reconnect.c	A sample program to reconnect to the cellular network
atop_4G_reset.c	A sample program to reset the cellular module

4.3.10 RTC Interface

There is one RTC clock via I2C interface, and it supports time unit to second/minute/hour/day/month/year up to year 2099.

5 Testing interfaces

ATOP provides some simple text programs. Please follow the below instructions to test the interfaces when it's needed.

5.1 Test COM port interface – transmit and receive

RS232/RS422/RS485 loopback test:

Execute rs232_loopback under the kernel shell. Be sure that you have connect the testing COM ports connected. Be sure to make TXD & RXD pins connected.

```
rs232_loopback
rs422_loopback
rs485_loopback
```

The baud rate is set at 115200

5.1.1 Test COM port interface by using atop_tcp_server

5.1.1.1 Test Method

The setup of the testing is shown as Figure 5.1.

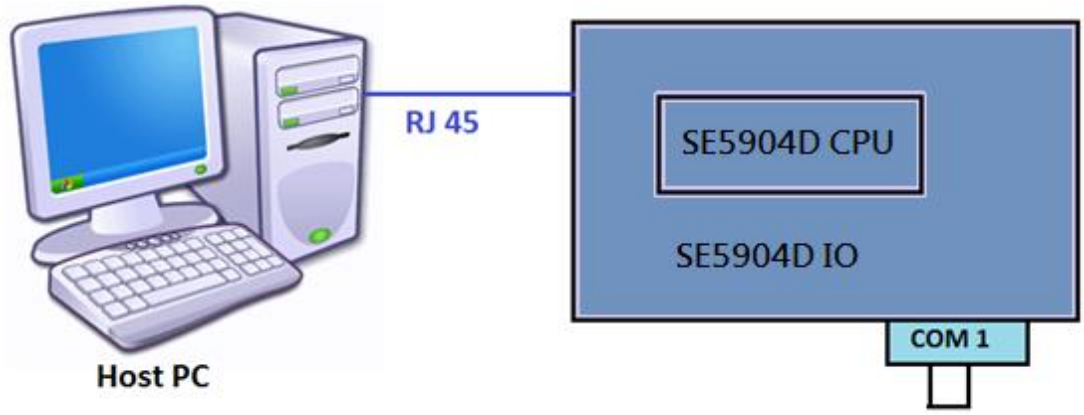


Figure 5.1 COM1 loopback test connection

5.1.1.2 Test Execution

1) Execute the command " atop_tcp_server " as next line to test RS232 with baud rate 115200.

atop_tcp_server RS232 115200 &

2) Type "ps -ef" from super terminal program to check if atop_tcp_server is executed as **Figure 5.2**

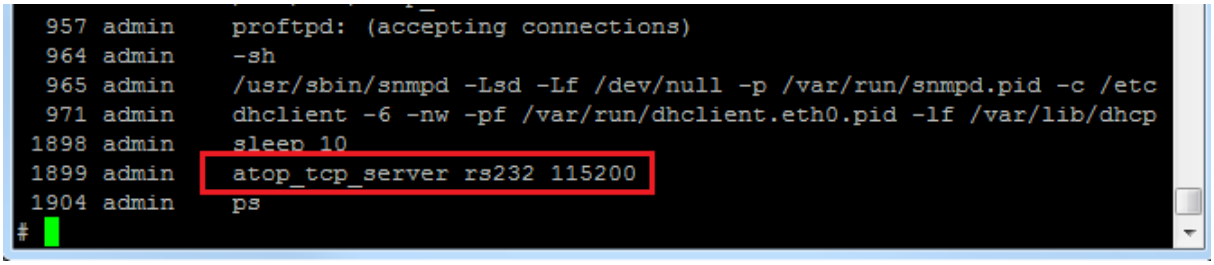


Figure 5.2 Process execution on SE5904D, example

3) Connect loopback for COM1 and execute TCPtest from MS-Windows as Figure 5.3

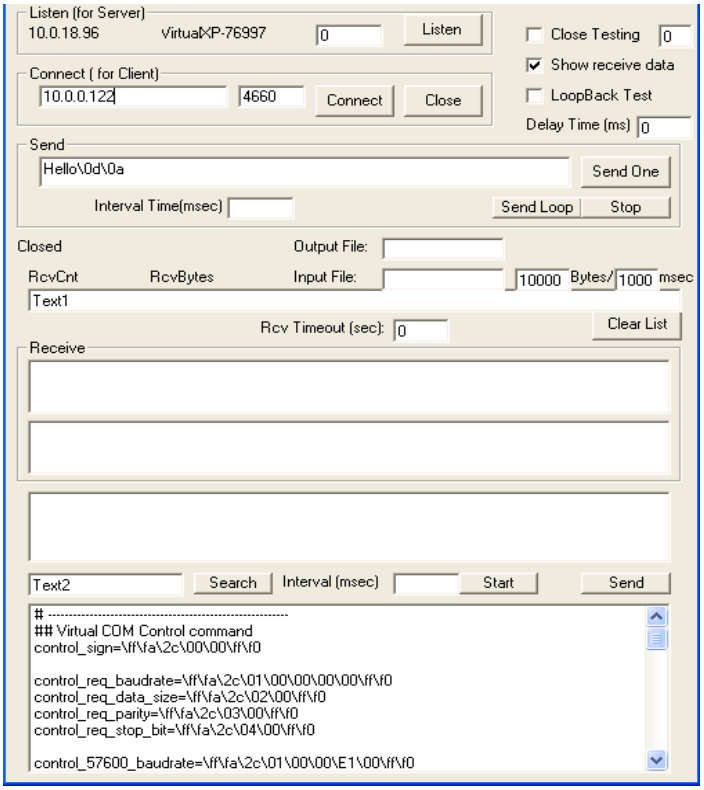


Figure 5.3 Setup TCPtest.exe for COM port loopback test

4) Select the TCP_Server mode and input IP address and TCP port number for COM1.
5) Click "Connect" to make TCP connections. The data keyed in "Send" box will be sent through the COM port. "Send Loop" is used to send continuously every certain period of time.
6) Click "Send One" to start the data transmission from PC to COM port. The data received from loopback link will be shown on the lower part "Receive" box of the Window as Figure 3-4.
7) You should be able to see "Hello" as hexadecimal display of each character "48 65 6C 6C 6F 0D 0A" shown on "Receive" box.

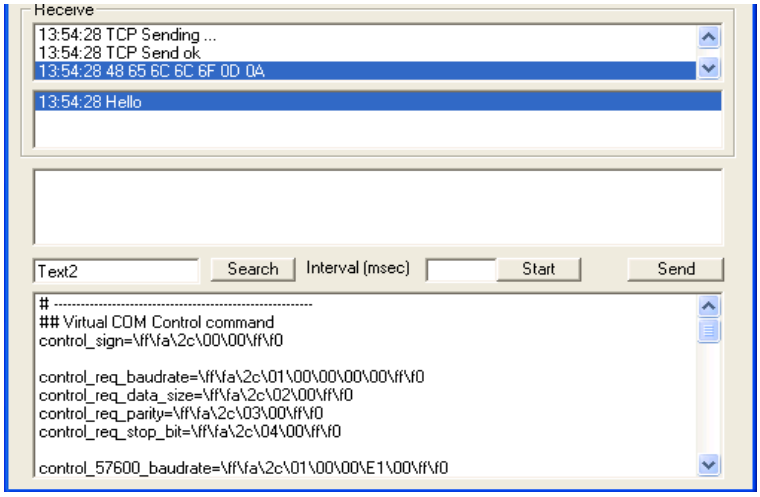


Figure 5.4 Result of loopback test

The default mapping table between TCP port number and COM port number:

Table 5.1 TCP-port to COM-port mapping

COM port	TCP port
COM 1	4660
COM 2	4661
COM 3	4662
COM 4	4663
...	...
COM 16	4660 + (16-1)

5.2 Test Buzzer interface

Upload the test file "buzzer" from ./software/atop_application folder into /jffs2 and execute the following command on the kernel shell:

```
./buzzer <on/off>
```
on: enable
off: disable

```
./buzzer on
```

You should hear the buzzer

5.3 Test Digital Input

Upload the test file "di_test" from ./software/atop_application folder into /jffs2 and execute the following command on the kernel shell:

```
./di_test
```

You can see the message print DI0/1 value.

5.4 Test Digital Output

Upload the test file "do_test" from ./software/atop_application folder into /jffs2 and execute the following command on the kernel shell:

```
./do_test
```

You can use multimeter to check the DO0/1 turn on 5 sec then turn off.

5.5 Test Hardware Relay Output

Use test tool "relay" to test HW relay device.

```
./relay
```

You can use multimeter to check the relay turn on then turn off after 10 sec.

5.6 Test Hardware Button

Use test tool "button" to get "press" then "release" event.

```
./button
```

5.7 Test Hardware Watchdog Interface (WDT)

Upload test file "hwd" from ./software/atop_application folder into /jffs2 and execute the following command on the kernel shell:

```
./hwd
```

If watchdog is not cleared or disabled in the source code, then system will restart automatically after 1.6 sec.

5.8 Test device LED

Upload test file "runLed" from ./software/atop_application folder to /jffs2 and execute the following command on the kernel shell:

```
./runLed <on/off>
```

on: enable
off: disable

You should see the RUN Led turn on or off.

Use test tool "alarmLed" to test HW alarm LED.

```
./alarmLed on
./alarmLed off
```

You should see the ALARM(Red) Led turn on or off.

5.9 Test RTC interface

Upload test file "rtc" from ./software/atop_application folder to /jffs2 and execute the following command on the kernel shell:

Set link file:

```
ln -s rtc get_rtc
ln -s rtc set_rtc
ln -s rtc rtc2system
ln -s rtc system2rtc
```

5.9.1 Setup RTC time:

Execute the following command on the kernel shell:

```
./set_rtc 2017/02/15-18:00:00
```

It will process both commands "date -s 2017-02-15 18:00:00" and "hwclock -w -f /dev/rtc1".

5.9.2 Read RTC time:

Execute the following command on the kernel shell:

```
./get_rtc
```

It will process command "hwclock -r -f /dev/rtc1".
The console will display the current RTC time such as "Wed Feb 1 14:11:50 2017"

5.9.3 RTC2system

Execute the following command on the kernel shell:

```
./rtc2system
```

It will process command "hwclock -s -f /dev/rtc1".
rtc2system: set system time from hardware clock.

5.9.4 system2RTC

Execute the following command on the kernel shell:

```
./system2rtc
```

It will process command "hwclock -w -f /dev/rtc1".
system2rtc: set hardware clock from system time.

In order to make sure the clock was set correctly, turn off the power and restart the system. After startup is completed, check the RTC time.

5.10 Using NOR Flash – JFFS2

There is a NOR flash on each device. 16MB of it is reserved for user applications mounted on /jffs2 file system. This will be mounted automatically on system start-up. The user can put all application programs and the related data into /jffs2.

All data in the /jffs2 will be kept when system is shut down.

5.11 MQTT

You can use http://test.mosquitto.org/ MQTT broker (server) for testing.

Subscriber and Publisher example:

```
mosquitto_sub -h test.mosquitto.org -t "atop" -v &
mosquitto_pub -h test.mosquitto.org -t "atop" -m"HelloWrold"
```

MQTT with example (You can download test certificates from test.mosquitto.org):

```
mosquitto_sub -h test.mosquitto.org -p 8883 -t "atop" --cafile /jffs2/mosquitto.org.crt &
mosquitto_pub -h test.mosquitto.org -p 8883 -t "atop" --cafile ./mosquitto.org.crt -d -m "test"
```

MQTT with username and password example:

```
mosquitto_sub -h192.168.4.238 -u atop -P 123456 -d -t atop &
mosquitto_pub -h192.168.4.238 -u atop -P 123456 -d -t atop -m"test123"
```

Please read https://github.com/mqtt/mqtt.github.io/wiki for details.

5.12 Firmware upgrade

Use test tool "fwmwr-upgrd" to upgrade kernel & rootfs.

```
./fwmwr-upgrd xxx.dld
```

note : The upgrade program only support dld file format.

6 Software API Reference

Software API is to be referred by the software application to configure system environment, include user name, password and network setting. The user can configure and then **restart** the system to make the new environment effective.

6.1 File List

Here is a list of all documented files with brief descriptions:

7 Data Structure Documentation

7.1 sessiontag Struct Reference

7.1.1 data fields

The documentation for this struct was generated from the following file:
atop_application/utils/atop_tcp_server/atop_tcp_server.c

- **pthread_t thandler**
- *int* **tid**
- *int* **uartfd**
- *int* **serv_sockfd**
- *int* **serv_acceptsockfd**
- *struct* **sockaddr_in serv_addr**
- *struct* **sockaddr_in client_addr**
- *int* **serv_link_state**
- *int* **serv_socket_init**
- *int* **err_count**

7.2 Network APIs

```
#include <stdio.h>
#include <fcntl.h>
#include <unistd.h>
#include <string.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include "atop_common.h"
#include "atop_eeprom.h"
#include "mobile.h"
```

7.2.1 Functions

void AtopSDKSetNetIP (int eth, char *ip)

AtopSDKSetNetIP

Sets the device's network IP on the selected Ethernet port

void AtopSDKSetNetMask (int eth, char *mask)

AtopSDKSetNetMask

Sets the device's subnet mask on the selected Ethernet port

void AtopSDKSetNetGateway (int eth, char *gw)

AtopSDKSetNetGateway

Sets the device's default Gateway IP on the selected Ethernet port

void AtopSDKSetNetDefGateway (int eth)

AtopSDKSetNetDefGateway

Sets the device's network IP on the selected Ethernet port

void AtopSDKSet4GDialOnBoot (int value)

AtopSDKSet4GDialOnBoot

Enables or disables automatic 4G dial on boot (SE5901B only)

void AtopSDKSet4GApn (char *apn)

AtopSDKSet4GApn

Sets the device's cellular Access Point (SE5901B only)

void AtopSDKSet4GPinEnable (char *pinCode)

AtopSDKSet4GPinEnable.

Sets the SIM card's PIN code (SE5901B only)

void AtopSDKSet4GPinDisable (void)

AtopSDKSet4GPinDisable.

Disables the SIM PIN code

void AtopSDKSet4GConnect (void)

AtopSDKSet4GConnect

Connects 3G/4G

void AtopSDKSet4GReConnect (int value)

AtopSDKSet4GReConnect

Sets automatic reconnect in case of connection loss

void AtopSDK4GHWReset (void)

AtopSDK4GHWReset

Resets the 3G/4G module

void AtopSDKSet4GDisconnect (void)

AtopSDKSet4GDisconnect

Disconnects from the 3G/4G Cellular network

7.3 Network APIs Function documentation

7.3.1 void AtopSDK4GHWReset (void)

API: AtopSDK4GHWReset

Arguments: nothing (void)
Function: resets 3G/4G module
Returns: nothing (void)
Example code:

```
218 {
219     int fno;
220     int value;
221     fno = open("/dev/atop_3g_reset", O_WRONLY);
222     if (fno > 0)
223     {
224         value = 0;
225         write(fno, &value, 1);
226         sleep(1);
227         value = 1;
228         write(fno, &value, 1);
229     }
230     else
231         printf("error (%d)\n", fno);
232     close(fno);
233     //printf("reset 3g module\n");
234     return;
235 }
```

*7.3.2 void AtopSDKSet4GApn (char * apn)*

API: AtopSDKSet4GApn
Arguments: apn (char) – string of APN (access point name)
Function: writes APN string to Module ini file
Returns: nothing (void)
Example code:

```
145 {
146     set apn info((uint8 t *)apn);
147 }
```

7.3.3 void AtopSDKSet4GConnect (void)

API: AtopSDKSet4GApn
Arguments: nothing (void)
Function: triggers a connection
Returns: nothing (void)
Example code:

```
190 {
191         threeg_connect();
192 }
```

7.3.4 void AtopSDKSet4GDialOnBoot (int value)

API: AtopSDKSet4GDialOnBoot
Arguments: value (integer)
Function: enables or disables 3G/4G cellular dial-on-boot function
Returns: nothing (void)
Example code:

```
129 {
130     set_dial_when_bootup(value);
131 }
```

7.3.5 void AtopSDKSet4GDisconnect (void)

API: AtopSDKSet4GDisconnect
Arguments: nothing (void)
Function: closes a 3G/4G cellular connection and resets the 3G/4G cellular module
Returns: nothing (void)
Example code:

```
247 {
248     threeg disconnect();
249     AtopSDK4GHWReset();
250 }
```

7.3.6 void AtopSDKSet4GPinDisable (void)

API: AtopSDKSet4GPinDisable
Arguments: nothing (void)
Function: disables 3G/4G cellular module SIM PIN check
Returns: nothing (void)
Example code:

```
176 {
177     set pin enable(0);
178 }
```

*7.3.7 void AtopSDKSet4GPinEnable (char * pinCode)*

API: AtopSDKSet4GPinEnable
Arguments: new PIN code (char)
Function: writes SIM PIN code to the 3G/4G cellular module to the ini file
Returns: nothing (void)
Example code:

```
161 {
162     set_pin_enable(1);
163     set pinCode((uint8 t *)pinCode);
164 }
```

7.3.8 void AtopSDKSet4GReConnect (int value)

API: AtopSDKSet4GReConnect
Arguments: recon value (int)
Function: writes 3G/4G cellular reconnection settings to the ini file
Returns: nothing (void)
Example code:

```
204 {
205     set_reconn_enable(value);
206 }
```

7.3.9 void AtopSDKSetNetDefGateway (int eth)

API: AtopSDKSetNetDefGateway
Arguments: eth: index of the Eth interface (ethX) - (int)
Function: writes default gateway settings to EEPROM
Returns: nothing (void)
Example code:

```
110 {
111     char defGW[16];
112
113     sprintf(defGW, "%d", eth);
114     AtopEESetDefaultGW(NULL, defGW);
115 }
```

*7.3.10 void AtopSDKSetNetGateway (int eth, char *gw)*

API: AtopSDKSetNetGateway
Arguments:

- eth: index of the Eth interface (ethX) - (int)
- gw: gateway address (char)

Function: writes gateway address settings to EEPROM
Returns: nothing (void)
Example code:

```
88 {
89     struct in_addr inp;
90     char info[16];
91
92     sprintf(info, "%dN", eth);
93
94     if (inet_aton(gw, &inp) > 0)
95         AtopEESetLanGateway(info, inet_ntoa(inp));
96 }
```

7.3.11 void AtopSDKSetNetIP (int **eth***, char* ***ip)***

API: AtopSDKSetNetIP
Arguments:

- eth: index of the Eth interface (ethX) - (int)
- ip: new IP address of port ethX (char)

Function: writes new IP address settings of specified port to EEPROM
Returns: nothing (void)
Example code:

```
34 {
35     struct in addr inp;
36     char info[16];
37
38     sprintf(info, "%dN", eth);
39
40     if (!strncmp("DHCP", ip, strlen("DHCP")))
41     {
42         AtopEESetLanIPMode(info, "DHCP");
43     }
44     else
45     {
46         if (inet_aton(ip, &inp) > 0)
47         {
48             AtopEESetLanIPMode(info, "Static");
49             AtopEESetLanIP(info, inet_ntoa(inp));
```

```
50          }
51      }
52  }
```

7.3.12 void AtopSDKSetNetMask (int eth*, char* *mask)

API: AtopSDKSetNetMask
Arguments:

- eth: index of the Eth interface (ethX) - (int)
- mask: subnet mask of port ethX (char)

Function: writes subnet mask settings to EEPROM
Returns: nothing (void)
Example code:

```
66  {
67      struct in_addr inp;
68      char info[16];
69
70      sprintf(info, "%dN", eth);
71
72      if (inet aton(mask, &inp) > 0)
73          AtopEESetLanNetmask(info, inet_ntoa(inp));
74  }
```

7.4 EEPROM User Name and Password Settings APIs

atop_application/utils/atop_libsdk/system.c File Reference

#include <stdio.h>
#include <fcntl.h>
#include <unistd.h>
#include <string.h>
#include "atop_common.h"
#include "atop_eeprom.h"

7.4.1 Functions

void AtopSDKSetUserName (char *name)

AtopSDKSetUserName

Saves the device's Administrator Username to the EEPROM

void AtopSDKSetUserPassword (char *password)

AtopSDKSetUserPassword

Saves the device's Administrator Password to the EEPROM

7.5 EEPROM User Name and Password Settings API Function documentation

7.5.1 void AtopSDKSetUserName (char **name***)*

API: AtopSDKSetUserName
Arguments: name – new administrator username (char)
Function: writes new System User Name to the EEPROM
Returns: nothing (void)
Example code:

```
27 {
28     char info[16];
29
30     sprintf(info, "N");
31
32     AtopEESetSysUserName(info, name);
33 }
```

7.5.2 void AtopSDKSetUserPassword (int **password***)*

API: AtopSDKSetUserPassword
Arguments: password (char)
Function: writes new password to the EEPROM
Returns: nothing (void)
Example code:

```
45 {
46     char info[16];
47
48     sprintf(info, "N");
49
50     AtopEESetSysUserPassword(info, password);
51 }
```

7.6 Run LED API Function documentation

include/atop_runled.h File Reference

7.6.1 Functions

Void AtopSetRunLed (u32 onMs, u32 offMs)

AtopSetRunLed

Sets RUN Led ON and OFF time settings

7.6.2 void AtopSetRunLed (onMs u32,offMs u32)

API: AtopSetRunLed
Arguments:

- onMs: delay on in ms - (u32)
- offMs: delay off in ms - (u32)

Function: sets RUN Led ON and OFF time settings
Returns: nothing (void)
Example code: not available

7.7 Alarm LED API Function documentation

include/atop_alarmled.h File Reference

7.7.1 Functions

void AtopSetAlarmLed (u32 value)

AtopSetAlarmLed

Turns the Alarm LED Off and On

7.7.2 void AtopSetAlarmLed (u32 **value)**

API: AtopSetAlarmLed
Arguments: value – on or off (u32)

- **0:** set to OFF
- **1:** set to ON

Function: Turns the Alarm LED ON or OFF
Returns: nothing (void)

7.8 Read Reset Button API

include/atop_button.h File Reference

7.8.1 Functions

int AtopButton (void)

AtopButton

Reads reset button status

7.8.2 int AtopButton (void)

API: AtopButton

Arguments: nothing (void)
Function: Reads reset button status
Returns: button status (int)

- **0**: button not pushed
- **1**: button pushed

7.9 Use Buzzer API documentation

include/atop_buzzer.h File Reference

7.9.1 Functions

void AtopBuzzerOnOff (int value)

AtopBuzzerOnOff

Starts or stops the buzzer

7.9.2 Macros

#define **BUZZER**_OFF 0
#define **BUZZER**_ON 1

7.9.3 void AtopBuzzerOnOff (int **value***)*

API: AtopBuzzerOnOff
Arguments: value – whether the buzzer should be on or off (0=OFF; 1=ON)
Function: starts or stops the buzzer
Returns: nothing (void)

7.10 Read Digital Inputs API documentation

include/atop_di.h File Reference

7.10.1 Functions

int AtopGetDI (int index)

AtopGetDI

Returns Digital Input value of DI index

7.10.2 int AtopGetDI (int index*)*

API: AtopGetDI
Arguments: index – defines the number of Digital Input to read value from
Function: Reads digital input value (high or low)
Returns: Digital Input value (int)

7.11 Write Digital Output API documentation

include/atop_do.h File Reference

7.11.1 Functions

int AtopSetDO (int index, int value)

AtopSetDO

Writes to Digital Output channel index, the value

7.11.2 int AtopSetDO (int index*, int* value*)*

API: AtopSetDO
Arguments:

- index: defines the number of Digital Output selected for control
- value: defines the status set to the selected Digital output (1 or 0)

Function: Controls digital output to be off or on
Returns: error code (int)

7.12 Hardware Watchdog API documentation

include/atop_hwd.h File Reference

7.12.1 Functions

Void atop_hwd_enable (void)

atop_hwd_enable

enables Hardware Watchdog

Void atop_hwd_disable (void)

atop_hwd_disable

disables Hardware Watchdog

Void atop_hwd_clear (void)

atop_hwd_clear

clears Hardware Watchdog

7.12.2 Macros

#define **HWD_DEV** "/dev/atop_hwd0"

7.12.3 void atop_hwd_clear (void)

API: atop_hwd_clear
Arguments: nothing (void)
Function: kicks hardware watchdog timer (reset)
Returns: nothing (void)

7.12.4 void atop_hwd_disable (void)

API: atop_hwd_disable
Arguments: nothing (void)
Function: disables Hardware watchdog
Returns: nothing (void)

7.12.5 void atop_hwd_enable (void)

API: atop_hwd_enable
Arguments: nothing (void)
Function: enables hardware watchdog control
Returns: nothing (void)

7.13 Wi-Fi USB Dongle control APIs documentation

These APIs can be used when SE59XX is used with one of the Approved USB to Wi-FI dongles. Please contact ATOP for more information on the supported models.

include/atop_libwl.h File Reference

7.13.1 Functions

Void create_default_ini_file (void)

create_default_ini_file

creates Wi-Fi INI configuration file. To be done before establishing a new connection

Int get_ssid (char **ssid)

get_ssid

reads the current SSID from the INI file and saves to **ssid

Int set_ssid (char **ssid_name)

set_ssid

sets new SSID to the INI file

Int get_key_mgmt (char **key_mgmt)

get_key_mgmt

reads the security configuration from the INI file and saves to **key_mgmt

Int set_key_mgmt (int mode)

set_key_mgmt

writes the security configuration to the INI file

Int get_psk (char **psk)

get_key_mgmt

reads the security key from the INI file and saves to **psk

Int set_psk (char **psk_name)

set_key_mgmt

writes the new security key to the INI file

Int parse_setting (void)

parse_setting

defines the parse settings

Void run_connection (void)

run_connection

Initiates the connection

7.13.2 void create_default_ini_file (void)

API: create_default_ini_file
Arguments: nothing (void)
Function: creates default INI configuration file for Wi-Fi dongle settings
Returns: nothing (void)
Example code:

```
13 {
14     FILE *fout;
15     fout = fopen(ini file name, "w");
16     fprintf(fout,
17     "\n"
18     "[network]\n"
19     "ssid = default;\n"
20     "key mgmt = WPA2-PSK;\n"
21     "psk = 12345678;\n");
22     fclose(fout);
23 }
```

7.13.3 void get_key_mgmt (char **key_mgmt***)*

API: get_key_mgmt
Arguments: key_mgmt – buffer for the management mode read value (char)
Function: reads Wi-Fi dongle key management mode from the INI configuration file
Returns: Error code (int)

- -1 : ERROR – Cannot open file; cannot read key management from file
- 0 : Successful

Example code:

```
86 {
87     char *str;
88
89     // read key from ini
90     dictionary *ini = iniparser load(ini file name);
91     if (ini==NULL) {
92         fprintf(stderr, "Cannot open %s\n",ini_file_name);
93         return -1;
94     }
95
96     // read key mgmt from ini
97     str = iniparser getstring(ini, "network:key mgmt", NULL);
98
99     // malloc a space and copy key_mgmt to it
100     if ( str!=NULL )
101         *key_mgmt = strdup(str);
102     else {
103         printf("Get key mgmt error!\n");
104         iniparser_freedict(ini);
105         return -1;
106     }
107
108     // free dictionary space
109     iniparser freedict(ini);
110     return 0;
111 }
```

7.13.4 int get_psk(char **psk***)*

API: get_psk
Arguments: psk : buffer for PSK (string)
Function: Reads current passphrase (PSK) from INI file and writes it to the argument
Returns: Error code (int)

- -1 : ERROR – Cannot open file, or get PSK error
- 0 : Successful

Example code:

```
157 {
158     char *str;
159
160     // read key from ini
161     dictionary *ini = iniparser load(ini file name);
162     if (ini==NULL) {
163         fprintf(stderr, "Cannot open %s\n",ini_file_name);
164         return -1;
165     }
166
167     // read psk from ini
168     str = iniparser_getstring(ini, "network:PSK", NULL);
169
170     // malloc a space and copy psk to it
171     if ( str!=NULL )
172         *psk = strdup(str);
173     else {
174         printf("Get PSK error!\n");
175         iniparser_freedict(ini);
176         return -1;
177     }
178
179     // free dictionary space
180     iniparser_freedict(ini);
181     return 0;
182 }
```

7.13.5 int get_ssid (char ssid*)*

API: get_ssid
Arguments: ssid : buffer for reading SSID (string)
Function: reads SSID from ini file.
Returns: Error code (int)

- -1 : ERROR: Cannot open file or function error
- 0 : Successful

Example code:

```
27 {
28     char *str;
29
30     // read key from ini
31     dictionary *ini = iniparser_load(ini_file_name);
32     if (ini==NULL) {
33         fprintf(stderr, "Cannot open %s\n",ini_file_name);
34         return -1;
35     }
36
37     // read ssid from ini
38     str = iniparser_getstring(ini, "network:SSID", NULL);
39
40     // malloc a space and copy ssid to it
41     if ( str!=NULL )
42         *ssid = strdup(str);
43     else {
44         printf("Get ssid error!\n");
45         iniparser freedict(ini);
46         return -1;
47     }
48
49     // free dictionary space
50     iniparser freedict(ini);
51     return 0;
52 }
```

7.13.6 int parse_setting (void)

API: parse_setting
Arguments: nothing (void)
Function: resets Wi-Fi connection settings (via USB-dongle)
Returns: nothing (void)
Example code:

```
226 {
227     char *ssid,*psk,*key mgmt;
228
229     // read key from ini
230     dictionary *ini = iniparser_load(ini_file_name);
231     if (ini==NULL) {
232         fprintf(stderr, "Cannot open %s\n",ini_file_name);
233         return -1;
234     }
235
236     // read ssid.psk.key_mgmt from ini
237     ssid = iniparser_getstring(ini, "network:SSID", NULL);
238     psk  = iniparser_getstring(ini, "network:PSK", NULL);
239     key mgmt = iniparser getstring(ini,  "network:key mgmt", NULL);
240
241     // parse ini into config format
242     FILE *fout ;
243     fout = fopen(parse_conf_name, "w");
244     fprintf(fout,"network={\n");
245     fprintf(fout,"\tscan ssid=1\n");
246     if (ssid != NULL) fprintf(fout,"\tssid=\"%s\"\n", ssid);
247     if ((psk != NULL) && (strcmp(key_mgmt,"None") != 0)) fprintf(fout,"\tpsk=\"%s\"\n", psk);
248     if (key_mgmt  != NULL) fprintf(fout,"\tkey_mgmt=%s\n}\n", key_mgmt);
249     fclose(fout);
250     iniparser freedict(ini);
251     return 0;
252 }
```

7.13.7 void run_connection (void)

API: run_connection
Arguments: nothing (void)
Function: initiate a Wi-FI connection through the USB Wi-Fi dongle. It runs authentication via wpa_supplication.
Returns: nothing (void)
Example code:

```
256 {
257     char cmd[60];
258     system("killall wpa supplicant");
259     sprintf(cmd,"wpa supplicant -Dnl80211 -iwlan0 -c%s &",parse conf name);
260     system(cmd);
```

7.13.8 int set_key_mgmt (int **mode)**

API: set_key_mgmt
Arguments: mode: - (int)

- **0**: Disabled
- **1**: WPA-PSK
- **2**: WPA2-PSK

Function: sets security key management mode to INI file for Wi-Fi dongle connection
Returns: error code (int)

- -1 : Cannot open file
- 0 : Successful

Example code:

```
114 {
115     // read key from ini
116     dictionary *ini = iniparser load(ini file name);
117     if (ini==NULL) {
118         fprintf(stderr, "Cannot open %s\n",ini_file_name);
119         return -1;
120     }
121
122     // setting key mgmt mode by user input
123     switch (mode) {
124         case 0 :
125             iniparser set(ini, "network:key mgmt", "None");
126             break;
127
128         case 1 :
129             iniparser_set(ini, "network:key_mgmt", "WPA-PSK");
130             break;
131
132         case 2 :
133             iniparser_set(ini, "network:key_mgmt", "WPA2-PSK");
134             break;
135
136         default :
137             printf("Input error!\n");
138             printf("0 : Disable\n");
139             printf("1 : WPA-PSK\n");
140             printf("2 : WPA2-PSK\n");
141             break;
142     }
143
144     // write key mgmt into ini
145     FILE *fin = fopen("/jffs2/tmpFile", "w");
146     iniparser_dump_ini(ini, fin);
147     iniparser freedict(ini);
148
149     fclose(fin);
150     remove(ini_file_name);
151     rename("/jffs2/tmpFile", ini_file_name);
152     return 0;
153 }
```

7.13.9 int set_psk (char psk_name)

API: set_psk
Arguments: psk_name: string of psk to be saved in ini file - (char)
Function: writes psk to ini file
Returns: int:

- -1 : Error – cannot open file, Invalid command, wrong length
- 0 : Successful

Example code:

```
185 {
186     char *key mgmt;
187
188     // read key from ini
189     dictionary *ini = iniparser load(ini file name);
190     if (ini==NULL) {
191         fprintf(stderr, "Cannot open %s\n",ini_file_name);
192         return -1;
193     }
194
195     // cannot set psk, if key_mgmt is disable
196     key_mgmt = iniparser_getstring(ini,  "network:key_mgmt", NULL);
197     if ( strcmp(key mgmt,"None")==0 ) {
198         printf("Invalid commend!\nkey mgmt is disable\n");
199         iniparser freedict(ini);
200         return -1;
201     }
202
203     // write psk into ini, if it's lengh between 8 and 63
```

```
204     if ( (8<=strlen(*psk name)) && (strlen(*psk name)<=63) ) {
205         iniparser set(ini, "network:PSK", *psk name);
206         FILE *fin = fopen("/jffs2/tmpFile", "w");
207         iniparser_dump_ini(ini, fin);
208 
209         fclose(fin);
210         remove(ini file name);
211         rename("/jffs2/tmpFile", ini file name);
212     }
213     else {
214         printf("Set PSK error!\nPSK length must be between 8 and 63.\n");
215         iniparser freedict(ini);
216         return -1;
217     }
218 
219     // free dictionary space
220     iniparser_freedict(ini);
221     return 0;
222 }
```

7.13.10 int set_ssid (char ssid_name)

API: set_ssid
Arguments: ssid_name: string of SSID - (char)
Function: writes SSID to ini file
Returns: int:

- -1 : error – SSID length problem or cannot open file
- 0 : successful

Example code:

```
55 {
56     // read key from ini
57     dictionary *ini = iniparser_load(ini_file_name);
58     if (ini==NULL) {
59         fprintf(stderr, "Cannot open %s\n",ini file name);
60         return -1;
61     }
62 
63     // write ssid into ini, if it's lengh between 1 and 32
64     if ( (1<=strlen(*ssid name)) && (strlen(*ssid name)<=32) ) {
65         iniparser set(ini, "network:SSID", *ssid name);
66         FILE *fin = fopen("/jffs2/tmpFile", "w");
67         iniparser_dump_ini(ini, fin);
68 
69         fclose(fin);
70         remove(ini file name);
71         rename("/jffs2/tmpFile", ini_file_name);
72     }
73     else {
74         printf("Set SSID error!\nSSID length must be between 1 and 32.\n");
75         iniparser freedict(ini);
76         return -1;
77     }
78 
79     // free dictionary space
80     iniparser freedict(ini);
81     return 0;
82 }
```

Atop Technologies, Inc.

www.atoponline.com
www.atop.com.tw

TAIWAN HEADQUARTER:

2F, No. 146, Sec. 1, Tung-Hsing Rd,
30261 Chupei City, Hsinchu County
Taiwan, R.O.C.
Tel: +886-3-550-8137
Fax: +886-3-550-8131

ATOP CHINA BRANCH:

3F, 75th, No. 1066 Building,
Qingzhou North Road,
Shanghai, China
Tel: +86-21-64956231

ATOP INDIA OFFICE:

Abhishek Srivastava
Head of India Sales
Atop Communication Solution(P) Ltd.
No. 22, Kensington Terrace,
Kensington Rd,
Bangalore, 560008, India
Tel: +91-80-4920-6363
E-mail: Abhishek.S@atop.in

ATOP INDONESIA BRANCH:

Jopson Li
Branch Director
Wisma Lampung Jl.
No. 40, Tomang Raya
Jakarta, Barat, 11430, Indonesia
Tel: +62-857-10595775
E-mail: jopsonli@atop.com.tw

ATOP EMEA OFFICE:

Bhaskar Kailas (BK)
Vice President (Business Development)
Atop Communication Solution(P) Ltd.
No. 22, Kensington Terrace,
Kensington Rd,
Bangalore, 560008, India
Tel: +91-988-0788-559
E-mail: Bhaskar.k@atop.in

ATOP AMERICAs OFFICE:

Venke Char
Sr. Vice President & Head of Business
11811 North Tatum Blvd, Suite 3031
Phoenix, AZ 85028,
United States
Tel: +1-602-953-7669
E-mail: venke@atop.in