

LOC

LinuxOnChip

User' Manual



LINUX Co., Ltd.



Understanding of LOC

These days people are very concerned about how to use Internet technology and Embedded Linux. Enterprises have much more interest in how to use these related technologies.

The uses of Internet technology and Embedded Linux have the following purposes:

1. The purpose of product operation and functional implementation changes the viewpoint of technology such as information-oriented product, flexibility of information transfer, horizontal information transfer, information management, etc. Moreover, various stratum require operations and uses of information that utilize the whole range. According to trends, people require general and successful network configurations that can integrate layers with the flexibility of information transfer.

The proper network is currently considered to be the Internet, and people have an interest in the latest technology because they can use various technologies in the environment.

2. Product development using a commercial operating system makes development difficult because of high cost and obstructive technology. People are, therefore, more concerned about the application of open and standardized Linux architecture. Because of its powerful functions but small capacity, Linux moves to the technology that can be used in embedded products and its application is increasing.

Because Linux includes TCP/IP, Linux is the perfect solution for Internet appliance. Since TCP/IP was developed in the Unix environment and Linux was developed in the Unix environment, Linux can present the most enhanced answer compared with the other solutions of Internet appliance.

Even though Linux is open, it still has many problems for embedded product applications, and is not easily applied because of a lack of experts and necessary technology. Because most solutions for Embedded Linux are based on PC architecture, the implementation of embedded type products is difficult due to cost and size.

For this reason, Linudix supplies the Linux-mounted CPU chip so that the functions can be easily implemented by enterprises that need Embedded Linux. LinuxOnChip (LOC) implements



these functions with its small size and low cost. The LOC-adopted enterprise can easily develop a product that has all the functions, only by developing applications of hardware and software. LOC is the easiest solution for products needing Internet interaction as well as Embedded Linux.

Product Configuration

- LOC
 - . CPU chip of 900mil DIP type
 - . Linux chip with internal CPU, ROM, RAM, and Ethernet controller

- CD for development environment configuration
 - . GCC m68k cross compiler
 - . GDB debugger
 - . Samples

- Evaluation board
 - . Evaluation board with fundamental H/W applications internally for LOC functional test

Contact

- E-mail: linudix@LinuxOnChip.com
- Telephone: +82-2-3482-0991~2
- FAX: +82-2-3482-0993

Fundamental Considerations for LOC Use

- For the use of Linux applications, you must have a fundamental knowledge of Linux. This is needed to implement applications based on an operating system.
- You must be familiar with C language. Linux is based on C language whose development tool is the C compiler. Most developers are familiar with this language.
- For the implementation of Internet interaction programming and home pages, you must have knowledge of these technologies. Linudix supports these technologies and you can find many examples throughout the Web and technical manuals for your needs.
- We will also answer your questions through the Q/A bulletin board column of our home page.



LinuxOnChip Features

- Optimal solution to use Linux in Embedded products
- Use in electric/electronic products needed to connect Internet such as Internet information household electric appliances, industrial controllers, etc.
- Implementation of real time kernel (Interrupt driven control possible)
- TCP/IP protocol stack
- File system in flash ROM
- Storage and execution possible of development program through Network File System (NFS)
- Implementation of on-line (target board) debugging
- LAN controller built-in
- LCD controller built-in
- Web server built-in
- PPP connections

H/W Description

Components	Description
CPU	Motorola MC68EZ328
ROM	2MB, 4MB flash memory
RAM	8MB DRAM
Ethernet port	10base-T
Interrupt pin	2 external pin
etc	2 external chip select LCD controller
Type	900mil width, 2.54mm pitch, 50pin DIP
Power	3.3V
Serial port	1 UART 1 SPI

CPU

The CPU is a Motorola MC68EZ328, 68000 series. So called Dragon ball, the CPU is a highly integrated one with a variety of peripheral devices; has a structure proper for usage in embedded systems; and its power consumption is very low.



Moreover, the CPU is a central processing unit whose performance is adequate enough to be used in embedded systems. For the configuration of applications or hardware using the LOC, visit the Motorola site to download and/or refer to the user' s manual.

** Note: High speed CPU products such as Strong APM are going to be released soon.*

Flash ROM

The flash ROM contains the Linux kernel. The kernel is programmed into the flash ROM at shipment. The flash ROM is classified as two types: 2Mbytes and 4Mbytes.

Because the kernel occupies around 1Mbyte, user area is 1Mbyte and 3Mbytes respectively.

The LOC contains flash file system, so user programs can be loaded in the user area.

Three partitions are basically mounted on the LOC.

/ (root): Root partition which is used to store main program and environment setting files of the LOC.

/var (Ram disk): Volatile RAM Disk which is usable for temporary storage.

/usr: User partition which install user applications and setting files, etc.

The size of each partition depends upon the capacity of the flash ROM. The following capacities are the actual ones usable after formatting:

2M bytes flash ROM

/ : 428KB

/usr: 620KB

/var: 500KB

4M bytes flash ROM

/ : 428KB

/usr: 2660KB

/var: 500KB

Note: The RAM disk capacity can be changed into 128K, 512K, 1M, 2M, or 4M depending upon necessity.

For the storage of flash files, refer to the flash file of Appendix.



DRAM

The DRAM is an area into which the kernel image is copied from flash ROM after LOC booting. Totally it has 8Mbytes, which are used for the main memory of the system – kernel, allocated variables, etc.

Ethernet controller

The Ethernet controller is 10base-T Ethernet controller. Users can have 10Base-T port only by installing the external transformer and connector. For recommendations, refer to “the Evaluation board in the Appendix”.

Serial port

There is only one UART port. The LOC has no RS-232C level converter by itself. Users can have a RS-232C port, installing the DSUB 9-pin connector in the form as required and level converter chip such as MAX232 in the external LOC. For installing, refer to “the Evaluation board in the Appendix”. The built-in serial port is also used as a console port for development.

SPI port (Serial Peripheral port)

The SPI port connects all kinds of peripheral chips through serial bus. The SPI is bus capable of interfacing peripheral devices with the CPU through three signal lines, and capable of connecting A/D, D/A, UART, etc. In case the SPI port is not used as it is, it can be used as a general I/O port. For more information, refer to “the user' s manual of MC68EZ328”.

Interrupt port

Two interrupt ports are supported. For interrupt priority, refer to “the user' s manual of MC68EZ328”. In case the interrupt pin is not used, it can be used as a general I/O port. Refer to the relationship between pin No. of CPU and pin No. of LOC, refer to “the Pin Assignment of Appendix”.

External chip select pin

Two Chip Select pins are provided for users to connect memory devices from the external LOC. A user does not need an additional address decoder circuit, and can use the two Chip Select signals as they are. For the relationship between pin No.' s of CPI and LOC, refer to “the Pin assignments in the Appendix”.

External memory bus

For the external memory bus, a total of ten address lines are supported. The external memory



access can be done up to 1024bytes per Chip Select. In most built-in product circuit design, a large address space is used for system ROM and RAM; and the actual peripheral does not occupy much space. Because the LOC has the internal big capacity of ROM and RAM, an external large space is not needed for address.

The data bus to the outside supports 8-bit. This resulted from the limit of pins according to the LOC configuration, but it is enough to use the external circuit only with 8 bits considering the features of peripheral circuits.

The upper 8 bits (D8 ~ D15) of CPU s data bus are provided for the external peripheral devices, and therefore, refer to “the Precautions of data bus use in Appendix”.

** Note) It is planned to release a product that has all the address space supported by MC68EZ328 as well as the 16-bit external data bus.*

LCD controller & 8 general I/O port

Eight general I/O ports are supported. This port and the built-in LCD controller share functions. In case the LCD controller is not used, it can be used as an I/O pin.

The built-in LCD controller supports 640 x 512 pixels for B&W and 320 x 240 for gray level. The LCD controller can be implemented without additional hardware. For more information, refer to “the user s manual of MC68EZ328”.

3.3V operating voltage

In addition to the adopted MC68EZ328, recently most CPUs uses 3.3V. Furthermore, more compactly designed CPUs prefer 3.3V. The LOC is also driven by 3.3V, and the external pin runs at 3.3V too.

Most users will be worried about the interface with applications circuit being driven by 5V. The interface of 3.3V and 5V circuitry is fundamentally as follows:

- Output: Because of TTL level, you have only to use TTL IC (74LSxx) in the receiving part of 5V.
- Input: Because the rated 3.3V is loaded on the input pin, 5V should not be loaded. The simple way is to use LVC or LVT series buffer, but if you have any cost trouble, you can solve it with using the voltage divider.
- Bidirectional port: Input or output problem can be simply solved, but it is comparatively complicated to connect bidirectional pins such as the data bus between 3.3V and 5V sources. By using the tri-state buffer along with input and output that are separated



respectively, it can be solved but the circuit is coarse. In this case, we think the best way is to use LVT245-like components. For products and description of 3.3V to 5V interfaces such as LVC or LVT, you can easily find out application note or product description at the Web site of Motorola or TI.



Development Environment Configuration

- Requirements for Development Equipment

It is the Linux-mounted desktop PC that is required for software development with LOC. Most PCs can be used as development equipment because they have CD-ROM drives or enough memory basically.

The CD contains:

- . Kernel uClinux 2.0.38
- . GCC m68k (Motorola 68000) compiler tool
- . Source for 68000 family support
- . GDB debugging tool
- . A variety of samples

Since the CD has been prepared based on the Redhat Package Manager (RPM), xx86 series RPM should be installed on the PCs that are used for development. If Linux has not been installed in your PC, you have only to install Red Hat Linux for Distribution.

- Configuration of m68k Development Environment

On the development PC, Users make, compiles, stores programs into the LOC to implement functions. Therefore, you have to prepare environment so that m68k compiler can work on the development PC.

Linux should be installed on the development PC, and it should be Red Hat 5.2 or higher version. If you installed the recent Linux for distribution, there may be no problem. You can check for the development tools compatible with Linux installed on your development platform as follows:

```
# ldd /bin/l
```

```
libc.so.6 => /lib/lib.so.6 (xxxx)
```

If “libc.so.5 => /lib/lib.so.5 (xxxx)” is displayed, you have to reinstall it.

First, insert the provided CD-ROM.

```
#mount -t iso9660 /dev/cdrom/mnt
```



Make a directory for development. In this example, the directory is /home/develop.

```
#mkdir /home/develop
```

Install the compiler.

```
#cd /mnt/cdrom/RPMS
```

```
#make
```

Now, the compiler is installed on the host PC. For the uses of m68k cross compiler, refer to “the on-line manual” .

```
#man m68k-pic-coff-gcc
```

Next, install the kernel source for LOC. If you need not to modify the Linux kernel, skip this step.

```
#cd /home/develop
```

```
#buildenv
```

```
#make
```

linux, romdisk, src directories are made in /home/develop directory when this step is finished. There will be uClinux 2.0.38 kernel source in the Linux directory, and will be system tools sources in the Src directory. The files lying on the Romdisk directory does not be currently used in the LOC.

Now, you have to change the original uClinux 2.0.38 kernel proper for the LOC hardware.

Execute as follows:

```
#cp /mnt/cdrom/LOC/LOC.diff.gz /tmp
```

```
#cd /tmp
```

```
#gzip -d LOC.diff.gz
```

```
#cd /home/develop/linux
```

```
#patch -p1 < /tmp/LOC.diff
```

(For configuration value that will be used to compile the kernel again in the future making config, refer to /mnt/cdrom/LOC/kernel_config.loc.)

Now, Linux source code for LOC has been installed.



From now, you can make, compile, and execute actual programs for the LOC.

Development Environment Test for Evaluation Board

Before testing the implemented development environment, let's see explanation first and test examples next of the evaluation board.

LinuxOnChip evaluation board

By using the LinuxOnChip evaluation board, you can test the function of LOC. It is an evaluation board that is used as a simple development tool. Connect the RS-232C port of PC that is equipped with terminal emulator program, to the RS-232 port of evaluation board and switch on the power. Then, the booting procedure of LOC is displayed on the terminal screen and the Login prompt is finally displayed. The LOC provides an interface such as desktop Linux, working as Linux machine.

Booting through Evaluation Board

1. Connect RS-232C cable between development PC and evaluation board – refer to “the Evaluation board of Appendix” for cable wiring.
2. Execute minicom on the development PC, and sets the environment of minicom to 9600bps, no parity, and 1 stop bit.
3. Switch on the power of 9 ~ 12V. The booting procedure and Login prompt is displayed in sequence. Login ID is “root” and the password is “uClinux” by default.
4. Fundamental Linux commands are built in. Test functions using the proper commands.

Test Example

Let's see the simplest example that displays "Hello! LinuxOnChip" on the console screen.

As soon as you build up the working environment in the directory, as you want in the host PC according to Configuration of m68k Development Environment mentioned above, the development environment is configured in the “development” directory. To avoid confusing the automatically produced files, let's create a “sample” working directory and make out a test code.

```
#pwd  
/home/xxx/development
```



```
#mkdir sample  
#cd sample  
#buildenv
```

At the end, make coding and save the following program.

```
main(int argc, char *argv[])  
{  
    puts("Hello! LinuxOnChip.");  
}
```

You have to compile after coding. You can do it with the following compile option.

```
#m68k-pic-coff-gcc test.c -o test
```

Now, you have made an executable file on the LOC, by the name of Test. You have to transfer this file to the LOC, but the file transmission can be done by the Network File System (NFS). You have to mount the working directory of the host PC on the LOC and from the viewpoint of LOC, set up a specific directory of the host PC so that you can use it as if it is a working directory of the LOC. Open “/etc/exports” file in the host PC and add data. Because you determined “/home/xxx/development/sample” directory as working directory in the example, add as follows:

```
/home/xxx/development/sample (ro,insecure)
```

Next, add IP address, assigned to the LOC, to the file “/etc/hosts” of the host PC as below – In this example, the IP address of LOC is 192.168.1.200; and that of the host PC is 192.168.1.7.

```
192.168.1.200 LOC.linux.private LOC
```

The NFS settings contain most Linux descriptions. At the end of “/etc/exports” file modification, execute as follows:

```
#/etc/rc.d/init.d/nfsserver stop  
#/etc/rc.d/init.d/nfsserver start
```



Now, Mount the working directory of host PC in the LOC in order to mount the disk of host PC on the LOC. Through this procedure, the disk of host PC can be used as the same as local disk—from the viewpoint of LOC.

From now, execute commands to the LOC through the console connected to Minicom, etc. Check for the current disk-mounted state, using the “mount” command.

```
#mount
/dev/root on / type umsdos (rw)
/dev/ram0 on /var type ext2 (rw)
proc on /proc type proc (rw)
/dev/vsbb on /usr type umsdos (rw)
```

Mount the “sample” directory of host PC to the “/mnt” directory of LOC, using the “mount” command.

```
#mount -t nfs 192.168.1.7:/home/xxx/development/sample /mnt
```

As soon as the “mount” command is successfully executed, check the result using the “mount” command.

```
#mount
/dev/root on / type umsdos (rw)
/dev/ram0 on /var type ext2 (rw)
proc on /proc type proc (rw)
192.168.1.7:/home/xxx/development/sample/mnt on /mnt type nfs (rw,addr=192.168.1.7)
/dev/vsbb on /usr type umsdos (rw)
```

As soon as the work ends as mentioned above, LOC can use the host PC as a local directory of itself. The IP address “192.168.1.7” is the host PC s.

Now, execute the program.

```
#cd /mnt
#./test
```



Hello! LinuxOnChip.

```
#
```

The result was printed out as respected. If you copy this executable file to the flash disk of LOC, this program is executed whenever the LOC is booted.

Follow the procedure below, and you can store this executable file to the flash disk – e.g. /usr/bin directory.

```
#cd /mnt
```

```
#cp test /usr/bin
```

```
#sync
```

Note: You have to use the sync command after copying the file to the flash memory. The sync command secures that the file has been copied. The operating system of Unix family such as Linux is designed to optimize the disk access speed with buffering – so called caching internally – about files. Therefore, enter sync command first after a cp command, and you may store the file to the flash disk. Consider sync command as a writing command to the flash ROM, and you can easily understand it.

If you have copied the file to the flash disk of LOC, you can execute this program even without host PC. If you want this program to be executed automatically whenever the LOC is booted, you have to add to the file “/usr/rc.d/rc” of LOC as follows”:

```
/usr/bin/test &
```

The file “/usr/rc.d/rc” is a shell script that is automatically executed after LOC has been booted. For users familiar with DOS, consider the function of the file as the same as that of autoexec.bat file.

For the modification of the file “/usr/rc.d/rc”, refer to the Flash file in the Appendix.



Appendix: DC characteristic

- Supply voltage: Min. 2.7V, Max. 3.6V
- Input voltage: Min. 2.7V, Max. 3.6V
- Operating temp: Min. 0°C, Max. 70°C
- Power rating: Max. 0.5W

3.3V input voltage should be kept in. Since the input voltage of chip in use is 3.3V, 3.3V or more may cause unexpected troubles.

Appendix: Memory map

	Address range
System RAM	0x00000000 ~ 0x00800000
System Flash ROM	0x11000000 ~ 0x11400000
External CSA*	0x11400000 ~ 0x11400800
Ethernet controller	0x10000300 ~ 0x11400800
External CSB*	0x10020000 ~ 0x10020800



Appendix: Flash file system

There is a file system on flash memory of LOC, so you may create, move, copy, or modify the file as if it has been saved on the hard disk.

Key in the “mount” command, and you can check the local disk allocated of LOC as follows -- / , /var, and /usr.

```
# mount
/dev/root on / type umsdos (rw)
/dev/ram0 on /var type ext2 (rw)
proc on /proc type proc (rw)
/dev/vsbb on /usr type umsdos (rw)
```

If you mounted the working directory of development PC by using the Network File System (NFS), the following is displayed on the screen:

```
# mount
/dev/root on / type umsdos (rw)
/dev/ram0 on /var type ext2 (rw)
proc on /proc type proc (rw)
192.168.1.7:/home2 on /mnt type nfs (rw,addr=192.168.1.7)
/dev/vsbb on /usr type vfat (rw)
```

You can know that the “/home2” directory of 192.168.1.7 host has been mounted. The LOC can use 192.168.1.7:/home2 directory as if it is its local directory.

Root directory has important files of the Linux system. There are LOC setting files under the “/etc” of the root directory.

/etc/rc : Shell script that is first executed when the LOC is booted.

/etc/rc.inet: Network setting file that sets up IP address, gateway, etc. Modify this file to change the network settings.

/etc/resolv.conf: Name server setting file. While using DNS, you have to set up the corresponding name server.

After modifying the flash disk file, you have to use the sync command to secure the writing operation of the actual flash disk.



#sync

When creating, removing, moving, or copying directory and file of “/” or “/usr” partition, you have to use the command mentioned above. In that case, if you do not execute the command, the file may not be written to the flash disk.

Note) After finishing program development on the LOC, you have to set the flash disk to Read-only mode. The function of flash disk in read-only mode is the same as that of general ROM, so system file and applications can be protected from damage due to a mistake.

Setting of flash disk to read-only mode is as follows:

1. Setting up the “/usr” directory to Read-only mode

Change the “/bin/mount /dev/vsbb /usr” line in the “/etc/rc” file as follows:

```
/bin/mount -o ro /usr
```

2. Setting up the “/” (root) directory to Read-only mode

Change the LOC into the Monitor mode. Refer to the “Changing into Monitor Mode” described below.

Key in Monitor mode as follows:

```
Mon>setenv cmdline Arg!ro
```

Boot again the LOC after execution is complete.

Changing into Monitor Mode

This is a method for setting the LOC into Monitor mode. If the ESC code is transmitted three times to the console port during LOC booting, Linux stops booting and starts Monitor mode. The simple way is to press the Reset button of LOC while you are pressing the ESC key on the terminal such as minicom, etc. – ESC code is continuously transmitted. As soon as “mon>” prompt appears, you need not to press the ESC key more because it means the Monitor mode is on.

One of the main functions of Monitor mode is the setting of environmental variables. Users can transmit the preset arguments to the kernel when Linux is booting. The environmental variable name is cmdline. In Monitor mode, you can enable or disable 60 environmental variables of LOC in maximum.



The command capable of setting the environmental variables is “setenv.” The command line is as follows:

```
setenv [variable name [value]]
```

Using the command without variable name and value displays the list of environmental variables that are currently stored in the LOC. Using a variable name without value deletes the variable. If variable name and value are both used together, the corresponding variable and value are stored in the LOC.

For example, “mon>setenv cmdline Arg!ro” makes an environmental variable of an cmdline whose value is set to Arg!ro; and “mon>setenv cmdline” makes the corresponding variable deleted. “mon>setenv” displays the current environment variables.

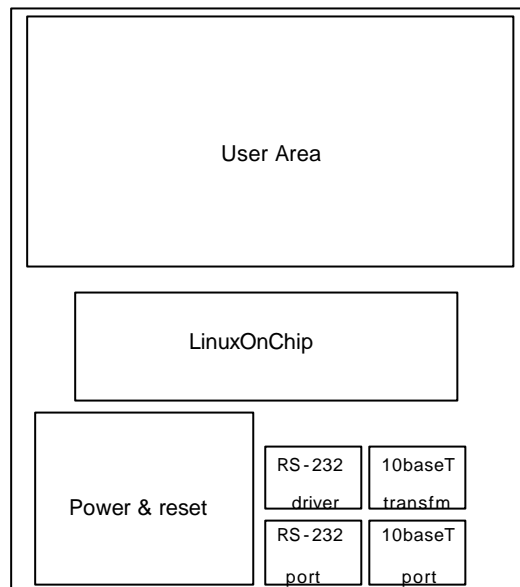
The cmdline, variable name, of environmental variable name is used to transfer arguments to Linux. A value should start with “Arg!”. It is the “ro” argument that sets the root disk to Read-only mode, of them transmitting arguments to Linux. For example, “mon>setenv cmdline Arg!ro” sets the root directory of LOC to Read-only mode.

Use “noprntk” if you don't want your system to print out the booting procedure to the serial port during Linux booting procedure. Use “noshell” argument if you don't want your system to activate the getty that displays Login prompt and stand by – this condition is needed when you want applications to control the serial port.

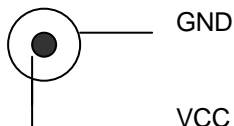
For example, “mon>setenv cmdline Arg!ro noprntk noshell” sets the root disk to Read-only mode without booting message. You can use this example when you do not want to use Login through serial port.



Appendix: Evaluation board



- Input Voltage: DC 9 ~ 12V



- 1 RS-232C serial port
- 1 10base-T Ethernet port
- Reset key
- Be sure of the direction of pin No. 1 when you mounting the LinuxOnChip (LOC) on the board.

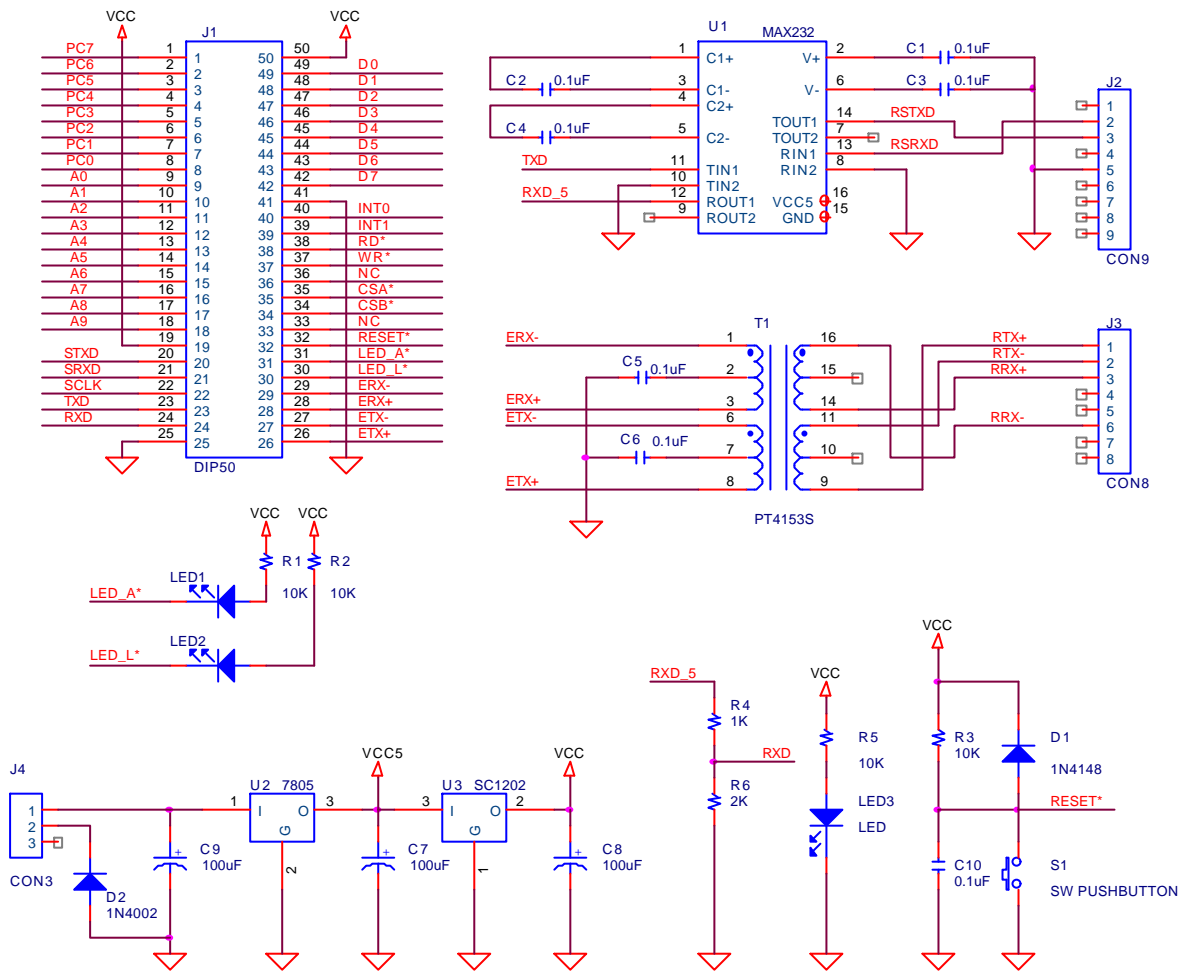
The No. 1 pin lies on the left bottom of Evaluation board, and has a hinge on the top of LOC case.

- 5V is loaded only on the RS-232C level converter, and the other parts use 3.3V.
- Transformer for 10base-T is used in 1:1 for receiving and 1:2.5 for transmitter side.

Transformers: TG92-2006N1 and TG41-2006N for Halo electronics, E2023 and EX2024 for Pulse engineering; and PT4153S for Valor electronics – integrated to the Halo.

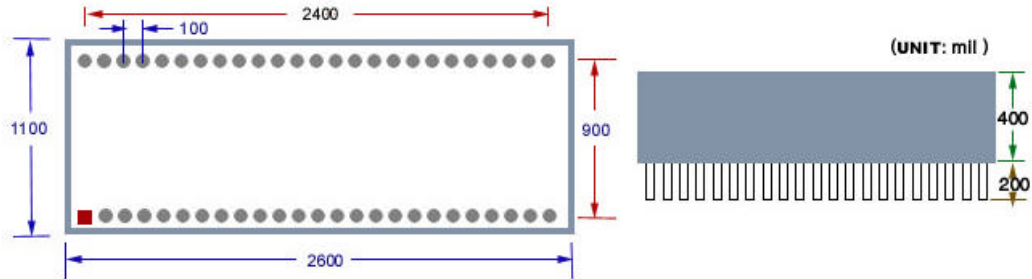


Circuit Diagram of Evaluation Board





Appendix: Dimension





Appendix: Pin Assignment and Description

CPU (MC68EZ328)		LinuxOnChip	
Pin number	Pin name	Pin number	Pin name
38,39,41 ~ 46	PC7 ~ PC0	1 ~ 8	PC7 ~ PC0
83 ~ 86,88 ~ 93	A1 ~ A10	9 ~ 18	A0 ~ A9
		19, 50	VCC
9	SPMTXD	20	STXD
10	SPMRXD	21	SRXD
11	SPMCLK	22	SCLK
14	TXD	23	TXD
13	RXD	24	RXD
		25, 41	GND
LAN controller I/O pin		26	ETX+
		27	ETX-
		28	ERX+
		29	ERX-
		30	LED L*
		31	LED A*
77	RESET*	32	RESET*
		33, 36	Don't care
54	CSB1*	34	CSB*
56	CSA1*	35	CSA*
80	UWE*	37	WR*
79	OE*	38	RD*
28	PD7	39	INT1
29	PD6	40	INT0
59 ~ 66	D15 ~ D8	42 ~ 49	D7 ~ D0

- Pins that have * just after its name mean low active.
- The address bus A[0:9] of LOC is exactly used as the A[1:10] of CPU, but the A0 (CPU side) has no meaning because it is accessed in 16 bits in the internal LOC. Users have only to connect the pin name A[0:9] of LOC to the A[0:9] of memory device when mounting the memory device on the external LOC.
- For the description of data bus, refer to "Considerations for Data Bus use."



Appendix: Considerations for Data Bus Use

The data bus D[15:8] of CPU is assigned to the D[7:0] of LOC. Be sure when you make out applications, referring to the following description. The data of memory devices connected to the D[7:0] of LOC are recognized as D[15:8] in the internal CPU.

The following is program output to show the difference when you access data bus in 16-bit or 8-bit. 0x1234 corresponds to D[15:0].

```
#Databus
Write word = 0x1234,      Read word    = 0x1200
Write word = 0x1234,      Read byte[0] = 0x12
Write word = 0x1234,      Read byte[1] = 0xa7
Write byte[0:1]=0x12,0x34, Read word    = 0x1200
Write byte[0:1]=0x12,0x34, Read byte[0] = 0x12
Write byte[0:1]=0x12,0x34, Read byte[1] = 0xa7
#
```

As shown above, 0x1234 is written in word. The word should be written as 0x1234, but 0x34 location is read to 0x00 because it is a program for the external 8-bit data. If it is read by a byte, a byte of No. 0 is read as 0x12 and a byte of No. 1 is read as 0x34. To the output, a byte of No. 1 is read to 0xa7, whose reason is arbitrary data is read because of only 8-bit data bus links to the outside. If you try to access to the internal 16-bit memory in the same way, it is read as 0x34. Likewise, if you try to access to the same address using the word after writing 0x12 to No. 0 byte and 0x34 to No. 1 byte in 8 bits, 0x1234 will be read.

These conditions are found in cases where Motorola CPUs are used, so most users who are familiar with Intel CPUs can get confused. For word write, word read, byte write, and byte read, users do not confuse it. However, for the access in byte read after word write, users have to be care of the conditions above in programming.

Note) The explanation mentioned above was done with an external memory device.



Appendix: Directory Structure

Directory structure and files stored in the flash disk of LOC are as follows:

1. Directory Structure

“/” directory has subdirectories as follows:

DOS: Empty directory. This directory is automatically produced in case the Umsdos file system is used.

Bin: Has fundamental utilities needed for the operation of LOC.

Dev: Has device files.

Sbin: Has executable files needed for a system.

Etc: Has the setting files of LOC system.

Lib: Has definition files for error values of library.

Usr: Storage directory of user's applications. The Usr is allocated to the flash disk partition independent of “/”.

Var: RAM disk-mounted directory. The RAM disk has a capacity of 512KB fundamentally.

Mnt: Directory to mount network drive.

Tmp: Directory to store temporary files, which is allocated to the RAM disk.

Proc: Has files of Linux kernel information.

Ramfs.img: The Image file of RAM disk, which is an empty disk image of 512KB.

2. “/usr” Directory Structure

The purpose of this directory is to store user's applications, and has a structure as follows:

/usr/bin: Users have only to copy applications to this directory.

/usr/rc.d: Has user's startup script. This directory has a file named rc. The rc file is shell script file that is executed just after system setting by itself at the end of LOC booting. Users have only to add a command to this file capable of starting applications.

3. Data for /etc Files

Users may be concerned about the system setting files that exist in the “/etc” directory. There are important files as follows:

/etc/rc: Shell script that is executed first after Linux booting. This script performs the system setting. In this script, “/etc/rc.inet”, the network setting script, is executed. But, users



need not to modify it. User startup shell script is “/usr/rc.d/rc” file.

/etc/rc.inet: Shell script responsible for the network setting. The setting of IP address subnet mask is executed in this shell script. Users can edit this file by hands, but can easily define network by using Netconfig command that produces “rc.inet” file automatically with their input data.

/etc/resolv.conf: DSN setting file. Users can easily edit this file by using Netconfig command.

/etc/inettab: Startup table starting the getty that receives Login after LOC booting. Users need not to modify this.

/etc/inetd.conf, */etc/services*: This file is needed for network demon named “inetd”. Users need not to modify this.

/etc/passwd: Password setting file, which is not referred by LOC. There is the hard-coding root passwd of LOC has been in the “/bin/login” executable file. Therefore, if you want to change password, you have to edit the login program source first in the development platform; to recompile it; and replace “/bin/login” with it.



Appendix: LOC-related Directory within Development Platform (Desktop PC) after Installation of LOC Development Environment

According to the procedures in the LOC main manual, your installation of tools existing in the development kit CD-ROM produces the following, and the installed main files are compiler tools, kernel source, and the sources of system utilities.

The files are all installed in “/opt/uClinux”, the subdirectory.

The contents of the subdirectory “/opt/uClinux” are as follows:

/opt/uClinux/bin: Has the executable files of compiler tools.

/opt/uClinux/linux: Sources of Linux kernel mounted on the LOC.

/opt/uClinux/src: Directory that has the sources of system utilities mounted on the LOC.

/opt/uClinux/m68k-coff: Directory that has tools needed for the compiler (kernel compile – different from the development compiler for user applications).

/opt/uClinux/m68k-pic-coff: Has a directory structure needed for the compiler of user program. There are three directories: ./bin, ./include, and ./lib.

./bin: Has tools (executable files) for compiler, assembler, linker, and libraries. These executable files make their link to “/usr/bin” during installation. The names of development tools are as follows:

m68k-pic-coff-gcc: GNU C compiler

m68k-pic-coff-ld: Linker

m68k-pic-coff-as: Assembler

m68k-pic-coff-ar, m68k-pic-coff-nm, m68k-pic-coff-ranlib, m68k-pic-coff-strip: Tools to process libraries and object files. On-line manual is fundamentally installed. Refer to the manual of each executable file.

e.g.) man m68k-pic-coff-gcc



`./include`: Directory that has standard header files. For example, in case of `#include <stdio.h>`, the header file “`stdio.h`” means `/opt/uClinux/m68k-pic-coff/include/stdio.h`.

`./lib` : Directory that has standard libraries (`Libc.a`)

It is not installed during the procedure of LOC main installation, but CD-ROM contains program samples. The samples exist in the directory “`/cdrom mount point/LOC/examples`” .