

LKV-316A

SW 사용자 메뉴얼

Board Rev. 1.2
BSP Version 1.2.1.3

2009 년 03 월 04 일



알 림

여기에 실린 내용은 제품의 성능 향상과 신뢰도의 증대를 위하여 예고없이 변경될 수도 있습니다.

여기에 실린 내용의 일부라도 엘케이일레브의 사전 허락없이 어떠한 유형의 매체에 복사되거나 저장될 수 없으며 전기적, 기계적, 광학적, 화학적인 어떤 방법으로도 전송될 수 없습니다.

㈜엘케이일레브

경기도 성남시 중원구 상대원동 191-1 SKn 테크노파크 메가동 1306 호

LKV-316A 은 ㈜엘케이일레브의 등록상표입니다.

차 례

알 림	2
1. INTRODUCTION	6
1.1. LKV-316A 보드의 소개	6
1.2. 본 문서의 내용	6
2. BOOT PARAMETER	7
2.1. BOOT PARAMETER 정보	7
2.2. BOOT PARAMETER 입력 예	7
3. LKV-316A CONTROL METHOD.....	9
3.1. LKV-316A 보드 MEMORY MAP	9
3.1.1. NVRAM	10
3.1.1.1. NVRAM 의 기능	10
3.1.1.2. NVRAM Control Function	10
3.1.2. FLASH MEMORY	11
3.1.2.1. Flash Memory 의 기능	11
3.1.2.2. Flash Memory Control Function	11
3.1.2.3. Flash Memory R/W Function	12
3.1.3. SRAM	13
3.2. THE CONTENTS OF A LKV-316A 보드 IO CONTROL	13
3.2.1. WATCH DOG TIME OUT RESET CONTROL	13
3.2.1.1. Watchdog Time Reset 기능	13
3.2.1.2. Watchdog time control function	14
3.2.2. FRONT PANEL LED CONTROL METHOD	14
3.2.2.1. Front Panel LED Control Function	14
3.2.3. DIP SWITCH READ METHOD	14
3.2.3.1. Function Description	14
3.2.3.2. Dip Switch Read Function	14
4. LKV-316A SERIAL CONTROL METHOD.....	16

4.1. LKV-316A 보드 SERIAL DRIVER INITIALIZATION.....	16
4.2. vxWORKS IO SYSTEM	16
4.2.1. OPEN ()	16
4.2.2. CLOSE ()	17
4.2.3. READ ()	17
4.2.4. WRITE ()	17
4.2.5. IOCTL ()	18
4.3. XR16L788 DRIVER IO CONTROL METHOD	20
4.3.1. XR16L788 BSP SPECIFIC IOCTL	20
4.3.1.1. XR16L788 General Function.....	20
4.3.1.2. XR16L788 Auto Control IOCTL Function	20
4.3.1.3. XR16L788 Error Check IOCTL Function	21
4.3.2. XR16L788 RS485 CONTROL METHOD PROGRAM EXAMPLE	22
4.3.2.1. RS485 통신 test Example.....	22
<u>5. LKV-316A 보드 VMEBUS CONTROL METHOD.....</u>	<u>24</u>
5.1. LKV-316A 보드 VME MEMORY MAP	24
5.2. LKV-316A VME BUS ACCESS EXAMPLE.....	24
5.2.1. EXTENDED MODE STANDARD MODE ACCESS EXAMPLE	24
<u>6. 주의 사항</u>	<u>26</u>

그림 목차

그림 1. LKV-316A 보드 의 Boot Parameter 입력정보	8
---	---

표 목차

표 1. LKV-316A Memory Map	9
표 2. DS9034(RTC) Register Map	11
표 3. LKV-316A VME Bus Memory Map	24

1. Introduction

1.1. LKV-316A 보드의 소개

LKV-316A CPU board 는 VMEbus 용 board 로써 Motorola 사의 PowerPC core 가 내장된 embedded communication processor 인 MPC860 을 사용하여 고성능과 다기능을 가능하게 하였다. Front panel 을 통하여 1 개의 ethernet port 와 2 개의 통신용 serial port 와 console port 가 장착되어 있다. 통신용 port 는 RS232 또는 RS485 통신이 가능하다.

또한 TM-8 Transition Module 을 장착할 때 8 개의 RS232/RS485 serial 통신 port 를 추가할 수 있으며 TM-16 Transition Module 을 장착할 때 16 개의 RS232/RS485 serial 통신 port 를 추가할 수 있다. 또한 Transition Module 을 사용하지 않고 VMEbus P2 커넥터를 통해서 16 개의 RS232 serial port 통신을 할 수 있다. board 내에 ARCnet mezzanine board 를 장착하면 VMEbus 를 통해서 ARCnet 통신도 가능하다.

VMEbus 에서는 system controller 기능과 Master 기능, Slave 기능, Interrupt handler 기능을 가지고 있다. board 내에 32 MB SDRAM, 1 MB EPROM, 512 KB RTC/NVRAM, 4 MB Flash Memory, 1MB SRAM 을 가지고 있어서 다양한 application 에의 사용이 가능하도록 설계되어 있다. OS 로는 VxWorks 가 지원된다. 또한 Backplane 기능이 지원되며, Master 보드에서 1 개의 LEN 으로 Slave 보드를 VME 버스를 통해서 Booting 시킬 수 있습니다.

1.2. 본 문서의 내용

LKV-316A 보드는 보드 내에 32Mbyte 의 SDRAM, 4Mbyte 의 Flash Memory, 512Kbyte 의 RTC/NVRAM, 1Mbyte 의 EPROM 으로 구성되어 있다. 또한 Serial Port 3 개와 VMEBus Interface 가 구현되어 있어 Master/Slave 보드로 사용 가능하다. 본 문서 에서는 위와 같은 보드 구성에 따라 1 장에서는 LKV-316A 보드에 대한 소개 및 구성에 대한 내용이 수록되어 있으며, 2 장부터는 보드 사용자가 LKV-316A 보드 사용 시 Setting 해야 할 Boot Parameter 에 관해 자세한 설명이 되어 있다. 3 장에서는 본 보드에서 지원되는 다양 메모리 종류의 컨트롤 방법 및 예가 주어지며, 4 장에서는 Serial Port Setting 및 사용 방법에 대해서 소개되며 그에 관한 예가 주어진다.

2. Boot Parameter

2.1. Boot Parameter 정보

boot device	: cpm
unit number	: 0
processor number	: 0
host name	: hjahn
file name	: vxWorks
inet on ethernet (e)	: 220.76.45.97
host inet (h)	: 220.76.45.36
user (u)	: hjahn
ftp password (pw)	: 1234
flags (f)	: 0x0
target name (tn)	:
other (o)	:

위의 내용은 Boot Parameter 설정 예제이다. 먼저 “boot device”는 Ethernet 드라이버의 name 을 가리키고 보드에 따라 달라지며, Device Driver 의 구성도 또한 다르다. File name 은 VxWorks 이미지가 있는 경로명을 적어준다. [그림 1]은 VxWorks Boot 를 하기위한 boot parameter 의 실제 입력 정보이다. inet on Ethernet(e)는 보드의 IP 주소를 적어주며, host inet 은 VxWorks 이미지를 다운로드 받을 host 의 주소를 적어준다. 다음으로 user(u)와 ftp password 에는 ftp demon 의 ID 와 Password 를 적어주면 된다. 여기서 Debug 용 Console 프로그램은 Tera Term 을 사용한다. BSP 는 FTP 를 이용하여 host 로부터 OS+Application image 를 다운로드 받으며, 후에 이를 실행한다.

2.2. Boot Parameter 입력 예

VxWorks Boot Prompt([VXWorks Boot]:) 상에서 ‘p’ 를 입력하면 [그림 1]과 같은 Boot Parameter 에 관한 정보를 볼 수 있다. 또한 Prompt 상에서 ‘c’ 를 입력하면 Boot Parameter 에 대한 정보를 사용자의 환경에 맞게 Setting 가능하도록 되어 있다. Boot Parameter 의 입력이 끝난 후 Prompt 상에서 ‘@’ 를 입력하면 Ethernet 을 통해 부팅이 실행되는 것을 확인 할 수 있다.

```
[VxWorks Boot]: @  
boot device       : cpm  
unit number      : 0  
processor number  : 0  
host name        : hjahn  
file name        : E:\ftp_root\vxworks\kvme316a\vxWorks  
inet on ethernet (e) : 220.76.45.97  
host inet (h)    : 220.76.45.36  
user (u)         : hjahn  
ftp password (pw) : 1234  
flags (f)        : 0x0  
  
Attached TCP/IP interface to cpm0.  
Attaching network interface lo0... done.  
Loading... 895764  
Starting at 0x40000..  
  
Target Name: vxTarget  
Attached TCP/IP interface to cpm unit 0  
Attaching network interface lo0... done.  
NFS client support not included.  
Loading symbol table from hjahn:E:\ftp_root\vxworks\kvme316a\vxWorks.sym ...done  
  
]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]  
]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]  
]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]  
   ]]]]]]]]]]  ]]]]  ]]]]]]]]]]           ]]]]]          (R)  
]   ]]]]]]]]]  ]]]]]]  ]]]]]]]]]  ]]]     ]]]          ]]]]  
]]   ]]]]]]]]  ]]]]]]]  ]]]]]]]]  ]]]     ]]]          ]]]]  
]]]   ]]]]]]  ]]]]]]  ]]]]]]]  ]]]]]]]]]  ]]]]]]  ]]]  ]]]]]]]  
]]]]   ]]]]  ]]]]]  ]]]]]  ]]]]]]]]]  ]]]]]]]  ]]]  ]]]]]]]  ]]]]]]  
]]]]]]  ]]]]]  ]]]]]  ]]]]]]]]]  ]]]]]]]  ]]]  ]]]]]]]  ]]]]]]]  ]]]]]]  
]]]]]]]]  ]]]]]]  ]]]]]]]  ]]]]]]]  ]]]]]]]  ]]]]]]]  ]]]]]]]  ]]]]]]]  
]]]]]]]]]  ]]]]]]  ]]]]]]]  ]]]]]]]  ]]]]]]]  ]]]]]]]  ]]]]]]]  ]]]]]]]  
]]]]]]]]]  ]]]]]]  ]]]]]]]  ]]]]]]]  ]]]]]]]  ]]]]]]]  ]]]]]]]  ]]]]]]]  
]]]]]]]]]  ]]]]]]  ]]]]]]]  ]]]]]]]  ]]]]]]]  ]]]]]]]  ]]]]]]]  ]]]]]]]  
]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]  
]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]  
]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]  
]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]  
Development System  
VxWorks version 5.4  
KERNEL: WIND version 2.5  
Copyright Wind River Systems, Inc., 1984-1999  
  
CPU: Etin System kvme316a. Processor #0.  
Memory Size: 0x2000000. BSP version 1.0/5.  
WDB: Ready.
```

그림 1. LKV-316A 보드 의 Boot Parameter 입력정보

3. LKV-316A Control Method

3.1. LKV-316A 보드 Memory Map

표 1. LKV-316A Memory Map

Memory range	Size(Byte)	Description
0x00000000 ~ 0x01FFFFFF	32M	SDRAM
0x10000000 ~ 0xEFFFFFFF	3.6G	VME Extended Address
0xF0000000 ~ 0xF0FFFFFF	16M	VME Standard Address
0xF1000000 ~ 0xF11FFFFF	1M	SRAM Space
0xF2000000 ~ 0xF201FFFF	128K	NVRAM Space
0xF3000000		RESERVED
0xF4000000 ~ 0xF400FFFF	64K	VME Short Address
0xF5000000		FAIL LED OFF
0xF5000002		FAIL LED ON
0xF6000000	1	DIP SW Read Address
0xF6000001		VMEbus Interrupt Request Status Read
0xF700000X		VMEbus Interrupt Acknowledge
0xF800000X		SCC0 Chip Select
0xF900000X		SCC1 Chip Select
0xFA00000X		ARCS Chip Select
0xFB000000		VMEbus Interrupt Vector Write Register
0xFB000001		VMEbus Interrupt request register
0xFB000002		RESERVED
0xFC000000		SCC Reset
0xFC000001		Watch Dog Timer Disable/Enable
0xFD000000 ~ 0xFD3FFFFFFF	4M	Flash Memory
0xFE000000		VMEbus RMC Cycle
0xFF000000		IMMR
0xFFF00000 ~ 0xFFFFFFFF	1M	EPROM

3.1.1. NVRAM

3.1.1.1. NVRAM 의 기능

NVRAM 은 Boot Parameter 의 저장을 주 목적으로 한다. 비휘발성 메모리로 Rom 은 아니지만 내부에 battery 가 내장되어 있어 특정 data 를 저장하기 위한 용도로 사용 가능하다. 또한 RTC 기능으로 사용 되며, NVRAM 에 Setting 한 시간 값에 의해 정확한 시간을 알아 낼 수 있다. 현재 LKV-316A 보드는 Dallas 사의 DS1646 를 사용하며, 메모리는 128Kbyte 이다. NVRAM 을 Access 할 경우 MPC860 의 특성으로 인하여 8bit 단위로 Access 되며, Read 시에는 16bit 나 32bit 단위로 Access 가능하다.

NVRAM 의 Access Range 는 0xF2000000 ~ 0xF201f000 까지 Access 가능하다.

현재 0xF201fdd0 부터 0xF201fed0 까지는 Boot Parameter 의 정보가 저장되어 있다.

3.1.1.2. NVRAM Control Function

1) STATUS timeSet(char *str)

RTC 의 시간정보를 Setting 하는 함수이다. Setting 값의 순서는 MM DD YY HH MM SS 의 순서로 설정된다.

Ex) timeSet("100504014800")	
MM	: Month
DD	: Day
YY	: Year
HH	: Hour
MM	: Minutes
SS	: Second

2) void timeShow(void)

현재 NVRAM 에 저장되어 있는 시간 정보를 출력해 주는 함수이다.

Ex) timeShow
WED AUG 27 11:30:04 2003

3) void rtcInit(base)

RTC 초기화 함수이다.

표 2. DS9034(RTC) Register Map

ADDRESS	DATA								FUNCTION	
	B7	B6	B5	B4	B3	B2	B1	B0		
1FFFF	–	–	–	–	–	–	–	–	YEAR	00–99
1FFFE	0	0	0	–	–	–	–	–	MONTH	01–12
1FFFD	0	0	–	–	–	–	–	–	DATE	01–31
1FFFC	0	FT	0	0	0	–	–	–	DAY	01–07
1FFFB	KS	0	–	–	–	–	–	–	HOUR	00–23
1FFFA	0	–	–	–	–	–	–	–	MINUTES	00–59
1FFF9	ST	–	–	–	–	–	–	–	SECONDS	00–59
1FFF8	W	R	S	–	–	–	–	–	CONTROL	A

ST : Stop Bit R : Read Bit FT : Frequency Test
W : Write Bit S : Sign Bit KS : Kick Start

3.1.2. Flash Memory

3.1.2.1. Flash Memory 의 기능

LKV-316A 보드의 Flash Memory 는 4Mbyte 의 용량을 가지고 있으며 flash Memory 를 이용하여 Flash Boot 기능으로 사용 가능 하다.

3.1.2.2. Flash Memory Control Function

1) makeFlashBoot();

Flash 에 bootrom.bin 파일을 플래쉬 메모리에 다운로드하는 함수이다.

LKV-316A 보드의 Flash Booting 을 하기 위해서는 아래와 같은 절차를 수행해야 한다.

순서 1) DOS 프롬프트 상에서 "WWINDBASEW\hostW\86-win32W\binW\torVars.bat"를 실행하여 Path 를 잡아준다.

순서 2) VxWorks bootrom.hex 파일을 Binary 형식으로 만든다. DOS 프롬프트 상에서 BSP 파일이 있는 디렉토리에서 다음을 실행한다.

```
Ex) make bootrom.hex
```

순서 3) 아래와 같은 명령으로 bootrom.bin 파일을 생성한다.

```
Ex) elfToBin <inFile> outfile_bsd
    elfToBin <bootrom> bootrom.bin
```

실행시 aout file 이 Binary 파일 형식으로 바뀜. 도스 명령의 가운데 인자는 롬파일 만들 때 생성된 bootrom 파일이다.

순서 4) 생성된 Binary 파일을 Target Board 에서 Load 할 수 있는 위치로 옮긴다.

순서 5) Console 상에서 "ls" 명령시 다음과 같이 binary 파일이 보이는지 확인한다

```
-> ls
bootrom.bin
```

순서 6) makeFlashBoot 함수를 이용하여 Binary 파일을 Flash Memory 에 기록한다.

순서 7) 아래의 makeFlashBoot Test 와 같은 메시지가 나오면 정상적으로 플래쉬 메모리에 기록된다. 아래 상태에서 롬의 우측 하단부에 있는 점퍼 2 개를 밑으로 옮기면 플래쉬 부팅이 이루어진다.

3.1.2.3. Flash Memory R/W Function

1) Flash Erase Function

- STATUS block_erase(int blknum)

blknum 에 해당되는 block 만을 지우는 함수이다. Block 의 개수는 32 개 이다.

– void all_erase(void)

flash Memory 전체를 clear 하는 function 이다. 현재 flash Memory 는 4Mbyte 이며, 전체 블록은 32 개의 블록을 가지고 있다.

2) flash Write Function

– STATUS flash_write(UINT32 addr, UINT16 *buf_pt, UINT32 buf_cnt)

현재 flash Memory 에 Write 하는 function 이다. 첫번째 함수 인자는 Flash Memory 의 Address 이다. 두 번째 인자는 Flash Memory 의 Write 될 Data 를 담고 있는 Buffer 의 첫 번째 pointer 가 된다. 마지막으로 세 번째 인자는 flash memory 에 Write 될 Buffer 의 Size 가 된다.

3) flash Read Function

– STATUS flash_read(UINT32 faddr, dst, UINT32 cnt)

현재 flash Memory 를 Read 하는 function 이다. 첫번째 함수 인자는 Flash Memory 의 Address 이다. 두 번째 인자는 Flash Memory 의 Read 될 Data 를 저장할 수 있는 Buffer 의 첫 번째 pointer 가 된다. 마지막으로 세 번째 인자는 flash memory 를 Read 할 Buffer 의 Size 가 된다.

3.1.3. SRAM

LKV-316A 보드의 SRAM 는 1Mbyte 의 용량을 가지고 있다. Address Range 는 0xF1000000 ~ 0xF11FFFFF 이다. SRAM 은 8bit, 16bit, 32bit 모두 Read/Write Access 가능하다

3.2. The contents of a LKV-316A 보드 IO Control

3.2.1. Watch dog time out reset control

3.2.1.1. Watchdog Time Reset 기능

LKV-316A Board 에서는 Maxim 사의 MAX690 칩을 사용하여 Watchdog time out 기능을 구현하였다. Watchdog Enable 함으로서 보드의 동작이 불안정시 다시 안정상태로 가능하게 할 수 있다(보드 reset).

3.2.1.2. Watchdog time control function

1) void watchDogTimer_Enable(void)

Watchdog timer Enable Function 이며, Watch Dog Timer Disable 시 Address 0xFC000001 에 0xFF 를 Write 하면 Watch Timer 가 Enable 이 된다.

2) void watchDogTimer_Disable(void)

Watchdog timer Disable Function 이며, Watch Dog Timer Enable 시 Address 0xFC000001 에 0x00 를 Write 하면 Watch Timer 가 Disable 이 된다.

3) STATUS watchDogTimer_Clear_Set(void)

Watchdog timer Setting Function 이며, 호출 시 Watchdog Timer 가 Clear 함으로서 Watchdog timer reset 이 걸리지 않게 된다. 실행 시 Diag1 LED 가 On/Off 되며, Timer 가 clear 된다.

4) STATUS watchDogTimer_Clear_Free(void)

함수 호출 시 Watchdog timer 는 clear 되지 않는다. 따라서 board 는 reset 된다.

3.2.2. Front Panel LED Control Method

3.2.2.1. Front Panel LED Control Function

1) STATUS ledCon(char led_value)

led_value 따라서 Diag LED 가 On/Off 된다.

EX) ledCon(1) : Diag0 LED ON
ledCon(2) : Diag0 LED OFF
ledCon(3) : Diag1 LED ON
ledCon(4) : Diag1 LED OFF

3.2.3. Dip Switch Read method

3.2.3.1. Function Description

LKV-316A Board 는 1 개의 Dip Switch 가 있으며, Switch On 시 data 는 0 으로 Read 되며, Switch Off 시 data 는 1 로 read 된다. DIP SWITCH READ Address 는 0xF6000000 이다.

3.2.3.2. Dip Switch Read Function

1) char DipSwValue(void)

Dip Switch Read 시 char 값이 return 되며, 값의 범위는 0x00 ~ 0xFF 사이의 값이다.

4. LKV-316A Serial Control method

4.1. LKV-316A 보드 Serial Driver Initialization

LKV-316A 보드는 1 개의 Ethernet Port 와 3 개의 Serial Port 가 전면에 장착되어 있으며, Serial Device 로는 "XR16L788" 2 개가 장착되어 있어 Serial Port 를 16channel 까지 지원한다. 또한 Baud rate generation 을 위한 기본 Clock 은 14.7456MHz 가 공급된다.

4.2. vxWorks IO System

LKV-316A 보드는 IO System 을 통해서 create(), remove() open(), close(), read(), write(), ioctl() 함수를 지원하며 vxWorks 에서 일반적으로 지원하는 기능과 같다.

4.2.1. open ()

1) Synopsis

```
int open ( const char * name, int flag, int mode )
```

2) Description

ⓐ name

현재 생성되어야 될 프로세서의 드라이버를 가리킨다. 드라이버의 이름은 각각의 프로세서마다 다르며 파일단위로 생성됨. Name 을 통해서 드라이버에 접근 가능함.

ⓑ flag

O_RDONLY(0) : 드라이버 읽기 전용으로 생성

O_WRONLY(1) : 드라이버 쓰기 전용으로 생성

O_RDWR(2) : 드라이버 읽기/쓰기로 생성

O_CREATE(0x0200) : 드라이버를 파일단위로 생성

ⓒ mode

UNIX 시스템에서 파일접근 허가를 나타내는 숫자를 수록함.

3) Return Value

ⓐ number

파일단위로 생성된 디바이스의 개수를 나타냄

ⓑ ERROR

해당 디바이스가 없거나 파일네임과 맞지 않을 경우를 나타냄

4.2.2. close ()

1) Synopsis

STATUS close (int fd)

2) Description

ⓐ fd

시스템으로부터 드라이버 파일의 사용이 끝났음을 알림.

3) Return Value

ⓐ OK

호출이 성공적으로 이루어 졌을 경우를 나타냄

ⓑ ERROR

해당 드라이버가 없을 경우 또는 파일 기술자가 아닐 경우.

4.2.3. read ()

1) Synopsis

int read (int fd, char * buffer, size_t maxbytes)

2) Description

개방된 드라이버 파일로부터 일정 수의 바이트를 버퍼로 복사하기 위해 사용됨.

ⓐ fd

버퍼로 읽어 들일 드라이버의 파일명

ⓑ buffer

읽어 들일 파일의 저장공간으로 char type 의 포인터로 정식 선언되며 1 문자와 1 바이트는 구별 없이 사용할 수 있다. 즉 buffer 는 자료가 복사되어질 문자배열에 대한 포인터 임.

ⓒ maxbytes

파일로부터 읽혀질 바이트의 수를 나타내는 양의 정수임.

3) Return Value

ⓐ number

파일단위로 생성된 디바이스에서 버퍼에 저장된 바이트의 개수를 나타냄

ⓑ ERROR

해당 디바이스가 없거나 파일네임과 맞지 않을 경우를 나타냄

4.2.4. write ()

1) Synopsis

int write (int fd, char * buffer, size_t nbytes)

2) Description

write 의 호출은 read 와는 반대로 문자배열로 선언된 프로그램 버퍼로부터 개방된 파일 드라이버를 통해 일정 수의 바이트를 출력하거나 쓰기 위해 사용됨

㉠ fd

출력하거나 쓰기 위한 드라이버의 파일명

㉡ buffer

문자배열로 선언된 프로그램 버퍼로 char type 의 포인터로 정식 선언되며 1 문자와 1 바이트는 구별 없이 사용할 수 있다 즉 buffer 는 출력될 자료의 문자배열에 대한 포인터 임.

㉢ nbytes

파일로부터 출력되어야 할 바이트의 수를 나타내는 양의 정수임.

3) Return Value

㉠ number

프로그램 버퍼로부터 디바이스에 쓰여진 바이트의 개수를 나타냄

㉡ ERROR

해당 디바이스가 없거나 파일네임과 맞지 않을 경우를 나타냄

4.2.5. ioctl ()

1) Synopsis

int ioctl (int fd, int function, int arg)

2) Description

Device 의 I/O 컨트롤 함수로 매우 유용하게 쓰인다.

㉠ fd

컨트롤 할 디바이스의 터미널 명

㉡ function

I/O 컨트롤 함수는 디바이스 드라이버에 따라 각 함수에 대한 옵션이 정해져 있으며 제공되는 드라이버의 옵션은 다음과 같다.

- FIOBAUDRATE

Baud Rate 을 Set 한다.

- FIOGETOPTIONS

각 채널에 대한 옵션을 얻어옴

- FIOSETOPTIONS

각 채널에 대한 옵션을 Setting 함

- FIOCANCEL

각 채널에 대한 read/write 요청을 거부함

- FIOFLUSH

RX Buffer clear

- FIOWFLUSH
TX Buffer clear

♣ VxWorks Programmer's Guide 3 장 참조

※ 주의 : 통신 도중에 에러가 발생해서 데이터 수신에 불안정한 경우에는 기본적으로 사용 중이던 포트를 close() API 를 이용해서 사용 중지하고, 다시 open() API 를 사용해서 해당 통신 포트를 초기화한 후에 read()/write() API 등을 사용해서 통신을 수행하는 것이 보다 안전합니다.

4.3. XR16L788 Driver IO Control Method

LKV-316A 보드는 IO System 을 통해서 create(), remove() open(), close(), read(), write(), ioctl() 함수를 지원하며 xr16l788 에서 제공되는 ioctl 의 parameter 는 다음과 같다.

4.3.1. XR16L788 BSP Specific IOCTL

4.3.1.1. XR16L788 General Function

XR16L788 에서 제공되는 드라이버 옵션은 다음과 같다.

- FIO_SET_BAUDRATE
Baud Rate 을 Setting 함
- FIO_GET_BAUDRATE
현재 channel 별로 Setting 되어 있는 BAUDRATE 의 값을 읽음
- FIO_R_FLUSH
Read Buffer Flush
- FIO_W_FLUSH
Write Buffer Flush
- FIO_GET_DREV
Driver 의 Reversion Number 을 읽어옴
- FIO_CHAN_RESET
Serial Channel 을 Reset 함
- FIO_SET_DATA_LEN
Channel 의 Data length 을 Set
- FIO_SET_STOP_BIT
Channel 의 Stop Bit 의 개수를 Set
- FIO_SET_PARITY
Channel 의 Parity Bit 를 Set(NONE, EVEN, ODD)
- FIO_SET_RTS_ON
- FIO_SET_RTS_OFF
- FIO_SET_DTR_ON
- FIO_SET_DTR_OFF

4.3.1.2. XR16L788 Auto Control IOCTL Function

1) XR16L788 Auto Control Function

XR16L788 에서 제공되는 드라이버 옵션은 다음과 같다.

- FIO_SET_FLOW_CTR
Channel Software Flow Control
- FIO_SET_AUTO_RS485CTR
Channel Auto RS485 Control Enable
- FIO_UNSET_AUTO_RS485CTR
Channel Auto RS485 Control Disable
- FIO_SET_AUTO_RTSDTR
Channel Auto RTSDTR Control Enable
- FIO_UNSET_AUTO_RTSDTR
Channel Auto RTSDTR Control Disable
- FIO_SET_AUTO_CTSDSR
Channel Auto CTSDSR Control Enable
- FIO_UNSET_AUTO_CTSDSR
Channel Auto CTSDSR Control Disable

4.3.1.3. XR16L788 Error Check IOCTL Function

1) XR16L788 Error Check Function

XR16L788 에서 제공되는 드라이버 옵션은 다음과 같다.

- FIO_RX_OVER_RUN_ERR
Channel Rx Over Run Error Number Check
- FIO_RX_PARITY_ERR
Channel Rx Parity Error Number Check
- FIO_RX_FRAMING_ERR
Channel Rx Framing Error Number Check
- FIO_RX_BREAK_ERR
Channel Rx Break Error Number Check
- FIO_RX_FIFO_ERR
Channel Rx FIFO Error Number Check

4.3.2. XR16L788 RS485 Control Method Program Example

LKV-316A 보드의 TM Module 에서 jumper 를 RS485 모드로 Setting 한다. (Setting 방법은 TM Module 하드웨어 설명서를 참조) 기본적으로 RS485 통신의 신호선은 RS232 와 별 차이가 없고 다만 물리적으로 하나의 신호선에 두 개의 라인이 필요한데 그들의 표현은 신호선명 뒤에 ‘+’ 와 ‘-’ 로 구분표기 한다. 하지만 UART 의 TXD, RXD 신호선이 멀티포인트 버스에 의하여 공동으로 사용하게 됨에 유의하여야 한다. 즉 하나의 마스터는 멀티포인트 버스를 출력이면 출력, 입력이면 입력으로 구분하여 사용할 수 밖에 없다. 일반적으로 RS485 통신시 개폐신호는 RTS 나 DTR 신호 중 하나를 사용하며, 시스템 베이스의 경우 RTS 신호를 사용한다.

4.3.2.1. RS485 통신 test Example

```
void tmRS485test(int from, int to, int loop_cnt)
{
    volatile char *buf;
    char *str="1234567890abcdef1234567890abcdef";
    int    k, r_byte, i=0;
    int    temp;

    printf("WnWnWnWnWn");
    printf("Wt*****Wn");
    printf("Wt*                               *Wn");
    printf("Wt*          TM RS485 LOOP Test          *Wn");
    printf("Wt*                               *Wn");
    printf("Wt*****Wn");
    printf("WnWn");

    /* 데이터 송신 포트 처리 */
    tmopenPort(from);
    ioctl(tmfp_port[from], 0xF44, 0);
    ioctl(tmfp_port[from], 0xF04, 10);
}
```

```

/* 데이터 수신 포트 처리 */
tmopenPort(to);
ioctl(tmfp_port[from], 0xF44, 0);
ioctl(tmfp_port[to], 0xF04, 10);

if( (from < 0 || from > 15) || (to < 0 || from > 15) ) {
    printf("Select correct port number : 0 ~ 15 \n");
    return;
}

while(1) {
    write(tmfp_port[from],str+i,16);      /* TX */
    r_byte=read(tmfp_port[to],buf,16);    /* RX */
    printf("r_byte = %d\n",r_byte);
    for (k=0; k<r_byte; k+=1)
        printf("%c");
    printf("\n");

    i++;

    if (i == 16)    i=0;
    /* 송신/수신 포트 바꿈 */
    temp = to;
    to = from;
    from = temp;

    if (loop_cnt != -1) {
        loop_cnt--;
        if (loop_cnt == 0)    break;
    }
}

/* 송신/수신 포트 close */
tmuartClose(from);
tmuartClose(to);
}

```

5. LKV-316A 보드 VMEBus Control method

5.1. LKV-316A 보드 VME Memory Map

표 3. LKV-316A VME Bus Memory Map

NAME	Address	Size(Byte)	Access	Description
VMEBUS Extended Address	0x10000000 ~ 0xEFFFFFFF	3.6G	R/W	VME A32 Access
VMEBUS Standard Address	0xF0000000 ~ 0xF0FFFFFF	16M	R/W	VME A24 Access
VMEBUS Short Address	0xF4000000 ~ 0xF400FFFF	64K	R/W	VME A16 Access

5.2. LKV-316A VME Bus Access Example

5.2.1. Extended Mode Standard Mode Access Example

```

void vme_RW_Test (unsigned int *extended_Adrs, unsigned int *standard_Adrs,
                  unsigned int vmeMSize, int flag)
{
    unsigned int i;
    unsigned int extendedSize, standardSize;
    unsigned int *extendedAdrs, *standardAdrs;
    int extendedData, standardData;
    unsigned int vme_loopCnt = 0;

    extendedSize = vmeMSize/4;
    standardSize = vmeMSize/4;

    while(1) {
        if (flag != -1) {
            flag--;
            if (flag == 0) break;
        }
    }
}

```



```

        extendedAdrs = extended_Adrs;
        printf("[%d]VME Extended Mode RW Test!!\n", vme_loopCnt);
        for (i=0; i<extendedSize; i++)
            *(extendedAdrs++) = 0;
        extendedAdrs = extended_Adrs;
        for (i=0; i<extendedSize; i++)
            *(extendedAdrs++) = i;

        extendedAdrs = extended_Adrs;
        for (i=0; i<extendedSize; i++) {
            extendedData = *(extendedAdrs++);
            if (i != extendedData)
                printf("Data Compare Error!! Extended Adrs[0x%x] : Original
Data[0x%x] : Error Data[0x%x]\n", extendedAdrs, i, *(extendedAdrs));
        }

        printf("[%d]VME Extended Mode RW Test OK!!\n", vme_loopCnt);
        standardAdrs = standard_Adrs;
        printf("[%d]VME Standard Mode RW Test!!\n", vme_loopCnt);

        for (i=0; i<standardSize; i++)
            *(standardAdrs++) = 0;

        standardAdrs = standard_Adrs;
        for (i=0; i<standardSize; i++)
            *(standardAdrs++) = i;
        standardAdrs = standard_Adrs;
        for(i=0; i<standardSize; i++) {
            standardData = *(standardAdrs++);
            if (i != standardData)
                printf("Data Compare Error!! Standard Adrs[0x%x] : Original
Data[0x%x] : Error Data[0x%x]\n", standardAdrs, i, *(standardAdrs));
        }
        printf("[%d]VME Standard Mode RW Test OK!!\n", vme_loopCnt++);
    }
}

```

6. 주의 사항



CAUTION

- VME Rack에서 가장 왼쪽에 위치한 보드는 System Controller로 설정할 것.
- System Controller 보드는 하나의 Rack에 하나만 장착할 것.
- System Controller 보드 이외의 보드는 마스터/Slave 보드로 설정할 것.
- KVME402 보드가 장착되는 Rack의 GND는 FGND와 직접 연결 하거나 Capacitor를 통해 연결할 것.



WARNING

- KVME402 보드는 정전기(Electrostatic Discharge)에 취약할 수 있으니, 보드 취급 시 주의할 것.
- KVME402 보드를 Rack에 장착할 시에는 가능하면 전원을 끄고 작업할 것.
- Rack이 접지되지 않았을 경우 감전의 우려가 있으므로, 반드시 접지 여부를 확인하고, 물이나 땀이 묻은 손으로 작업하지 말 것.



(주)엘케이일레븐

138-809 서울특별시 송파구 가락 2 동 545-5 번지 동명빌딩 3 층

<http://www.lk11.com>, 전화:02-3012-3788