SONY®

OPEN-R SDK

Revision Record

OPEN-R

20030210-E-003

Notes on This Document

Notes on Using This Document

- ❑ The contents provided by this document (PDF files) are intended only for supplying information.
- ❑ The contents provided by this document (PDF files) are subject to change without notice.
- ❑ We are not responsible for errors or omissions in technical or editorial aspects concerning the contents described in this document. We also are not responsible for technical measures, correspondence, execution according to this document, as well as for the results occurred by them such as inevitable, indirect or incidental damages.

Notes on Copyright

- ❑ Sony Corporation is the copyright holder of this document.
- ❑ No information in this document may be duplicated, reproduced or modified. It is also prohibited to publish the contents of this document on the Internet Website or other public media without the express written permission of Sony Corporation.

About Trademarks

- ❑ AIBO and OPEN-R are trademarks or registered trademarks of Sony Corporation.
- ❑ "Memory Stick" is a trademark of Sony Corporation. "TM" is not described in this document.
- ❑ Windows is registered trademark of Microsoft Corporation in the United States and/or other countries.
- ❑ Mac OS is registered trademark of Apple Computer, Inc in the United States and/or other countries.
- ❑ Linux is a registered trademark of Linus Torvalds.
- ❑ MIPS is a registered trademark of MIPS Technologies, Inc. in the United States and/or other countries.
- ❑ Acrobat and Adobe are registered trademarks of Adobe Systems Incorporated.
- ❑ Other system names, product names, service names and firm names contained in this document are generally trademarks or registered trademarks of respective makers.

Index

Revision Record

The revision record excludes the correction for tabs and indents.

20030210-E-003

The revision record (20030210-E-003) are written for Manuals (Old version).

The revision record for Installation Guide
Old version (20020603-E-001)
Old version (20020730-E-002)
Old version (20030201-E-003)
New version (20030210-E-004)

The revision record for Programmer's Guide
Old version (20020603-E-001)
Old version (20020730-E-002)
New version (20030201-E-003)

The revision record for Level2 Reference Guide
Old version (20020603-E-001)
Old version (20020730-E-002)
New version (20030201-E-003)

The revision record for Model Information ERS-210
Old version (20020603-E-001)
Old version (20020730-E-002)
New version (20030201-E-003)

The revision record for Model Information ERS-220
Old version (20020603-E-001)
Old version (20020730-E-002)
New version (20030201-E-003)

The revision record for OPEN-R Internet Protocol Version4
New version (20020603-E-001)

Installation Guide

Changes from 20030201-E-003 to 20030210-E-004

Page 4

1.2 Download files, OPEN-R SDK

Change:

OPEN_R_SDK-sample-1.1.3-r3.tar.gz
OPEN_R_SDK-docE-1.1.3-r3.tar.gz

To:

OPEN_R_SDK-sample-1.1.3-**r4**.tar.gz
OPEN_R_SDK-docE-1.1.3-**r4**.tar.gz

Page 5

2.1 Cygwin

The following are added.

In the rest of this document, we will unpack packages in cygwin's path by using its 'tar' command. In this explanation, whenever you see cygwin's path referred to(for example, /usr/local), it indicates cygwin's /usr/local, NOT c:\usr\local (from the root of your hard drive).

Page 6

The following are added.

2.2 CygIPC

1 Install CygIPC with the following steps. (/xxx is your directory where the downloaded file is placed)

```
cd /
tar zxvf /xxx/cygipc-1.13-2.tar.gz
```

Change:

2.2 gcc

To:

2.3 gcc

2.2 gcc

The following are deleted.

Here, we will unpack this package in the cygwin   fs /usr/local (which corresponds to C:\cygwin\usr\local on your hard drive). In the rest of this document, whenever you see /usr/local, it indicates cygwin's /usr/local, NOT c:\usr\local.

2.2 gcc, Step1

Change:

tar xzf /xxx/mipsel-devtools-3.2-bin-r1.tar.gz

To:

tar **zxvf** /xxx/mipsel-devtools-3.2-bin-r1.tar.gz

Page 7

Change:

2.3 OPEN-R SDK

To:

2.4 OPEN-R SDK

2.3 OPEN-R SDK, Step1

Change:

tar xzf /xxx/OPEN_R_SDK-1.1.3-r2.tar.gz

To:

tar zxvf /xxx/OPEN_R_SDK-1.1.3-r2.tar.gz

2.3 OPEN-R SDK, Step1

Change:

The directory /usr/local/OPEN_R_SDK/OPEN_R will be created.

To:

The directory /usr/local/OPEN_R_SDK/OPEN_R, **RP_OPEN_R** will be created.

2.3 OPEN-R SDK, Step2

The following are deleted.

The directory /usr/local/OPEN-R SDK/RP_OPEN_R will be installed.

Change:

2.4 Sample programs

To:

2.5 Sample programs

2.4 Sample programs, Step1

Change:

tar xzf /xxx/OPEN_R_SDK-sample-1.1.3-r3.tar.gz

To:

tar **zxvf** /xxx/OPEN_R_SDK-sample-1.1.3-r4.tar.gz

The following are deleted.

2.5 CygIPC
2 Install CygIPC with the following steps.

```
cd /
tar zxf cygipc-1.13-2.tar.gz
```

2.6 AIBO Built-in Flash ROM Upgrade, Step1

Change:

tar xzf /xxx/upgrade-OPEN_R-1.1.3-r2.tar.gz

To:

tar **zxvf** /xxx/upgrade-OPEN_R-1.1.3-r2.tar.gz

Change:

The directory Upgrade will be created.

To:

The directory **u**pgrade will be created.

2.6 AIBO Built-in Flash ROM Upgrade, Step2

Change:

under the directory Upgrade.

To:

under the directory **u**pgrade.

Page 8

3.1 Building

Change:

make PREFIX=/mydir/OPEN_R_SDK

To:

make **OPENRSDK_ROOT** =/mydir/OPEN_R_SDK

Page 12

4.1.2 Running on host, Step3

Change:

$ /usr/local/OPEN_R_SDK/RP_OPEN_R/bin/start-rp-openr

To:

/usr/local/OPEN_R_SDK/RP_OPEN_R/bin/start-rp-openr

Page 13

4.2.2 Distributed execution on AIBO and host, Step1

Change:

```
cd /xxx/sample/LMasterRSlave/RP/robot
```

To:

cd /**mydir**/sample/LMasterRSlave/RP/robot

Changes from 20020730-E-002 to 20030201-E-003

Page 4

1.2 Download files, OPEN-R SDK

Change:

OPEN_R_SDK-1.1.3-r1.tar.gz
OPEN_R_SDK-sample-1.1.3-r1.tar.gz
OPEN_R_SDK-docJ-1.1.3-r2.tar.gz
OPEN_R_SDK-docE-1.1.3-r1.tar.gz

To:

OPEN_R_SDK-1.1.3-**r2**.tar.gz
OPEN_R_SDK-sample-1.1.3-**r3**.tar.gz
OPEN_R_SDK-docJ-1.1.3-**r3**.tar.gz
OPEN_R_SDK-docE-1.1.3-**r3**.tar.gz

1.2 Download files, For windows platforms

Change:

cygwin-packages-1.3.10-bin.exe
mipsel-devtools-3.0.4-bin-r1.tar.gz

To:

cygwin-packages-**1.3.17**-bin.exe
mipsel-devtools-**3.2**-bin-r1.tar.gz

The following are added.

cygipc-1.13-2.tar.gz

1.2 Download files, Source files

Change:

cygwin-packages-1.3.10-src.tar.gz
binutils-2.12.tar.gz
gcc-3.0.4.tar.gz
newlib-1.9.0.tar.gz

To:

cygwin-packages-**1.3.17**-src.tar.gz
binutils-**2.13**.tar.gz
gcc-**3.2**.tar.gz
newlib-**1.10.0**.tar.gz

The following are deleted.

build-devtools-3.0.4-r1.sh	shell script for building the above three files

The following are added.

cygipc-1.13-2-src.tar.gz	Source files of cygpic

1.2 Download files

The following are added.

- ❑ Scripts

build-devtools-3.2-r1.sh
Shell script for building binutils, gcc and newlib

build-devtools-3.2-macosx-r1.sh
Shell script for building tools on Mac OS X

Page 5

2.1 Cygwin, Step1,2

Change:

cygwin-packages-1.3.10-bin.exe

To:

cygwin-packages-**1.3.17**-bin.exe

Page 6

2.2 gcc, Step1

Change:

/xxx/mipsel-devtools-3.0.4-bin-r1.tar.gz

To:

/xxx/mipsel-devtools-**3.2**-bin-r1.tar.gz

Change:

GNU binutils-2.12
GNU gcc-3.0.4
newlib-1.9.0

To:

GNU binutils-**2.13**
GNU gcc-**3.2**
newlib-**1.9.10**

Change:

build-devtools-3.0.4-r1.sh

To:

build-devtools-**3.2**-r1.sh

Change:

binutils-2.12.tar.gz
gcc-3.0.4.tar.gz
newlib-1.9.0.tar.gz

To:

binutils-**2.13**.tar.gz
gcc-**3.2**.tar.gz
newlib-**1.10.0**.tar.gz

2.3 OPEN- R SDK, Step1

Change:

tar xzf /xxx/OPEN_R_SDK-1.1.3-r1.tar.gz

To:

tar xzf /xxx/OPEN_R_SDK-1.1.3-**r2**.tar.gz

The following are added.

2 To install Remote Processing OPEN-R, run the setup-rp-openr script.
/usr/local/OPEN_R_SDK/RP_OPEN_R/bin/setup-rp-openr

The directory /usr/local/OPEN-R SDK/RP_OPEN_R will be installed.

2.4 Sample program, Step1

Change:

tar xzf /xxx/OPEN_R_SDK-sample-1.1.3-r1.tar.gz

To:

tar xzf /xxx/OPEN_R_SDK-sample-1.1.3-**r3**.tar.gz

The following are added.

2.5 CygIPC
Install CygIPC with the following steps.

```
cd /
tar zxvf cygipc-1.13-2.tar.gz
```

Change:

2.5 AIBO Built-in Flash ROM Upgrade

To:

2.6 AIBO Built-in Flash ROM Upgrade

Page 10

3.2.2 How to setup WLANCONF.TXT

Change:

Copy WLANDFLT.TXT in /OPEN-R/SYSTEM/CONF/ of an AIBO programming memory stick to WLANCONF.TXT, and edit it

To:

Copy WLANDFLT.TXT in /OPEN-R/SYSTEM/CONF/ of an AIBO programming memory stick to WLANCONF.TXT, and edit it **In OPEN-R SDK 1.1.3 r2, DNS SERVER support is available. This new feature is commented out by default**

Change:

```
APMODE =2
CHANNEL=3
```

To:

```
APMODE =2
CHANNEL=3
#DNS_SERVER_1=10.0.1.1
#DNS_SERVER_2=10.0.1.2
#DNS_SERVER_3=10.0.1.3
#DNS_DEFDNAME=example.net
```

The following are added.

DNS_SERVER_1,2,3

Specify IP addresses of up to three DNS servers

DNS_DEFDNAME

Specify default domain name

Page 11

The following are added.

Chapter4 Remote Processing OPEN-R

Notes

For communication between AIBO and remote host through a wireless LAN, it is necessary to boot AIBO before accessing it from the remote host.

4-1 Building & running ObjectComm

4-1-1 Running on AIBO

1 Build the executable file. (Here, mydir is an example directory)

```
cd /mydir/sample/ObjectComm
```

2 Copy these two OPEN-R directories to a blank AIBO Programming Memory Stick.

```
/usr/local/OPEN_R_SDK/OPEN_R/MS/WCONSOLE/nomemprot/OPEN-R
/mydir/sample/ObjectComm/MS/OPEN-R
```

3 Insert the AIBO Programming Memory Stick into AIBO, then boot it. Subsequent operations (usage of wireless console, how to shutdown AIBO, etc.) are the same as before.

4-1-2 Running on host.

1 Run ipc-daemon.

```
/usr/local/bin/ipc-daemon
```

2 Build the executable file.

```
cd /mydir/sample/ObjectComm/RP/host
make install
```

3 Run start-rp-openr

```
$ /usr/local/OPEN_R_SDK/RP_OPEN_R/bin/start-rp-openr
[pid:29444,msqid:196610,oid:0x00030002] oserviceManager
[pid:29445,msqid:229379,oid:0x00038003] tcpGateway
[pid:29446,msqid:262148,oid:0x00040004]
        MS/OPEN-R/MW/OBJS/SUBJECT.BIN
[pid:29447,msqid:294917,oid:0x00048005]
        MS/OPEN-R/MW/OBJS/OBSERVER.BIN
 SampleSubject::Ready() : ASSERT READY
SampleObserver::Notify() !!! Hello world !!!
SampleSubject::Ready() : ASSERT READY
SampleObserver::Notify() !!! Hello world again !!!
SampleSubject::Ready() : ASSERT READY
```

4 Type ctrl-c to terminate the program.

4-2 Building & running LMasterRSlave
4-2-1 Running on AIBO.
1 Build the executable file.

```
cd /mydir/sample/LMasterRSlave
make install
```

2 Copy the following two OPEN-R directories to a blank AIBO Programming Memory Stick.

```
/usr/local/OPEN_R_SDK/OPEN_R/MS/WCONSOLE/nomemprot/
OPEN-R
 /mydir/sample/LMasterRSlave/MS/OPEN-R
```

3 Insert the AIBO Programming Memory Stick into AIBO. Then boot AIBO. Subsequent operations (usage of wireless console, how to shutdown AIBO, etc.) are the same as before.

4-2-2 Distributed execution on AIBO and host

In this case, system objects and POWERMON.BIN are running on AIBO, while LMRS.BIN is running on the host.
0 Run ipc-daemon (for Cygwin only)

```
 /usr/local/bin/ipc-daemon
```

Procedures for AIBO
1 Build the executable file.

```
cd /xxx/sample/LMasterRSlave/RP/robot
make install
```

2 Copy the following OPEN-R directories to a blank AIBO Programming Memory Stick.

```
/usr/local/OPEN_R_SDK/OPEN_R/MS/WCONSOLE/nomemprot/
OPEN-R
```

3 Edit the following according to your wireless setup.

```
/OPEN-R/SYSTEM/CONF/WLANCONF.TXT
```

4 Insert the AIBO Programming Memory Stick into AIBO, then boot it. Subsequent operations (usage of wireless console, how to shutdown AIBO, etc.) are the same as before.

Procedures for host
1 Build the executable file.

```
cd /mydir/sample/LMasterRSlave/RP/host
make install
```

2 Edit the following.

```
MS/OPEN-R/MW/CONF/HOSTGW.CFG
```

Change 10.0.1.100 to the IP address of your AIBO.

```
!ROBOT_PROXY 59001 10.0.1.100
TCPGateway.Sensor.OSensorFrameVectorData.S 59002 10.0.1.100
TCPGateway.Effector.OCommandVectorData.O 59003 10.0.1.100
```

3 Run start-rp-openr

```
/usr/local/OPEN_R_SDK/RP_OPEN_R/bin/start-rp-openr
```

4 Type ctrl-c to terminate the program.

Notes rp-openr-ipcrm
SystemV IPC resources may sometimes remain unreleased after termination of the program. You can examine information about the SystemV IPC resources by using the ipcs command. Run the rp-openr-ipcrm script to release any remaining SystemV IPC resources.

```
/usr/local/OPEN_R_SDK/RP_OPEN_R/bin/
rp-openr-ipcrm
```

4-3 Limitations of Remote Processing OPEN-R

- The configuration of the system objects on AIBO is limited to 'nomemprot' when executing a program distributed between AIBO and a remote host. Therefore, please use the following directory as the OPEN-R directory to be copied to an AIBO Programming Memory Stick.

```
/usr/local/OPEN_R_SDK/OPEN_R/MS/WCONSOLE/nomemprot/
OPEN-R
```

- Description using [RobotDesign] as shown below is not supported in OBJECT.CFG and CONNECT.CFG on the host.

```
#
# OBJECT.CFG
#
[ERS-210]
/MS/OPEN-R/MW/OBJS/ERS-210.BIN

[ERS-220]
/MS/OPEN-R/MW/OBJS/ERS-220.BIN
```

- ❑ Unavailable OPEN-R API for a host program

  OPENR::ControlPrimitive()
  OPENR::NewSoundVectorData()
  OPENR::DeleteSoundVectorData()
  OPENR::NewCdtVectorData()
  OPENR::DeleteCdtVectorData()
  OPENR::SetCdtVectorData()
  OPENR::Shutdown()
  OPENR::ObservePowerStatus()
  OPENR::UnobservePowerStatus()
  OPENR::FindDesignData()
  OPENR::DeleteDesignData()
  OPENR::Fatal()

- ❑ ANT (Aperios Network Toolkit) class library is not available for a host program.

Changes from 20020603-E-001 to 20020730-E-002

Page 1

About Trademarks

Change:

Acrobat and Adobe is a registered trademark …

To:

Acrobat and Adobe **are** registered trademarks…

Page 4

1.2 Download files, OPENR-R SDK

Change:

OPEN_R_SDK-docE-1.1.3-r1.tar.gz Manuals

To:

OPEN_R_SDK-docE-1.1.3-**r2**.tar.gz Manuals

1.2 Download files, For ERS-210 users

Change:

upgrade-OPEN_R-1.1.3-r1.tar.gz

To:

upgrade-OPEN_R-1.1.3-**r2**.tar.gz

1.2 Download files, Notes

Change:

Each file includes …

To:

Each file **name** includes…

Page 7

2.5 AIBO Built-in Flash ROM Upgrade, Step1

Change:

tar xzf /xxx/upgrade-OPEN_R-1.1.3-r1.tar.gz

To:

tar xzf /xxx/upgrade-OPEN_R-1.1.3-**r2**.tar.gz

Page 9

3.2 Running, Step3

Change:

Insert an AIBO wireless LAN card into AIBO.

To:

Insert an AIBO wireless LAN card **and an AIBO Programming Memory Stick into AIBO. Then boot AIBO**.

3.2 Running, Step4

This step is deleted.

3.2 Running, Step6

The following is deleted.

Insert the AIBO Programming Memory Stick into AIBO and boot AIBO.

3.2 Running, Step6

The following is added.

Notes

There are two ways to connect your PC to AIBO: with a wireless access point, or with an additional wireless LAN adapter in ad-hoc mode. To make sure AIBO is successfully connected to your network, you can execute the following console command after booting AIBO.

```
ping (IP address of AIBO)
```

Programmer’s Guide

Changes from 20020730-E-002 to 20030201-E-003

Page 35

The following is added.

Chapter6 Remote Processing OPEN-R

Remote Processing OPEN-R is a remote processing environment where you can execute an OPEN-R based program on a remote host which is not AIBO.

By using Remote Processing OPEN-R, some objects can be executed on the remote host(connected to AIBO via wireless LAN), and other objects can be executed directly on AIBO. All objects will be executed as one program, distributed between the two machines.

Here are some advantages of Remote Processing OPEN-R:

- Objects executing on the remote host can reconnect to objects executing on AIBO without interrupting AIBO fs execution. So, you can shorten the turn-around time for coding, executing, and debugging, and develop a program efficiently.

- There is source code compatibility between AIBO's objects and remote host's objects. So, you can use rich debugging tools(e.g. gdb) on the remote host.

- While executing a program, you can use the rich resources and various functions that the remote host PC provides.

In Remote Processing OPEN-R, we use TCPGateway objects on AIBO and also on the remote host. TCPGateway is the OPEN-R object that implements communication between the objects executing on AIBO, and the objects executing on the remote host. The communication between AIBO's objects and the remote host's objects are done by passing the ordinary protocol of the OPEN-R inter-object communication over the wireless LAN.

There is source code compatibility between AIBO's objects and remote host's objects, but there is no binary compatibility.

Remote Processing OPEN-R gives you the environment for building native (x86) binary executable from source code that works on AIBO (by compiling with the OPEN-R SDK). The binary files that work on Remote Processing OPEN-R are not identical to the binary files that work on AIBO.

Page 38

A.2 C standard library

Change:

times

To:

time

Changes from 20020603-E-001 to 20020730-E-002

Page 7

2.2 Core class

Change:

As shown in figure3,

To:

As shown in figure **1-3**,

Page 8

2.2 Core class, (1), Destroy method

Change:

This is called at startup…

To:

This is called at **shutdown**…

Page 15

2.5 Sending and receiving data

Change:

… obsFunc1 and sbjFunc2 are IDs to find an observer and a subject, which are defined in def.h (which is automatically generated from stub.cfg by executing 'stubgen2').

To:

… obsFunc1 and sbjFunc2 are index numbers for the arrays observer[] and subject[], respectively, to identify the observer and the subject. They are defined in def.h, which is automatically generated from stub.cfg by executing the 'stubgen2' command.

Page 23

4.3.2 Execution of AIBO Programming Memory Stick, Notes

Change:

the remaining battery life…

To:

the remaining battery **capacity**…

Page 28

5.2.2 System operation at CPU exception

Change:

…on the memory stick…

To:

…on the **AIBO Programming Memory Stick**…

5.2.2.1 Default EmON.CFG

Change:

…on the memory stick…

To:

…on the **AIBO Programming Memory Stick**…

Page 30

Notes

Change:

If you specify an option –g in the compilation of an object, both the result of disassembling and source codes can be intermingled, by replacing the argument –d in mipsel-linux-objdump with –S.

To:

□ If you specify **the** option –g in the compilation of an object, both the result of disassembling and source codes can be intermingled by replacing the argument –d in mipsel-linux-objdump with –S.

Page 31

5.3 Commands in EMON.CFG

Change:

5.3 Commands in EMON.CFG

To:

5.2.5 Commands in EMON.CFG

Page 32

objs

Change:

You can look the relation…

To:

You can **look at** the relation…

Page 36

A1. File system

Change:

There is no difference between the file descriptors 0, 1, or 2.

To:

There is no difference **among** the file descriptors 0, 1, or 2.

Changes from 20020730-E-002 to 20030201-E-003

Page 67

OPENR::ControlPrimitive(), Description

The following are added.

```
oprmreqSPEAKER_SET_SOUND_TYPE
oprmreqSPEAKER_GET_SOUND_TYPE

/* Set sound data type */
OPrimitiveControl_SpeakerSoundType soundType
        (ospksndMONO16K16B) ;
OPENR : :ContorlPrimitive(speakerID,
        oprmreqSPEAKER_SET_SOUND_TYPE.
        &soundType, sizeof (soundType) );

/* Get sound data type */
OPrimitiveControl_SpeakerSoundType soundType;
OPENR : :ContorlPrimitive(speakerID,
        oprmreqSPEAKER_SET_SOUND_TYPE.
        &soundType, sizeof (soundType) );
```

Page 84

The following are added.

Chapter6 wireless LAN API

As for the details for the obtained data, refer to the header file of each data type or the sample program.

ERA201D1_GetMACAddress()

Syntax

```
EtherStatus ERA201D1_GetMACAddress
        (EtherDriverGetMACAddressMsg* msg)
```

Description

This gets the MAC address.

Parameters

msg MAC address

Returned value

ETHER_ OK	Success
ETHER_INVALID_PORT	No WLAN card exists.
ETHER_UNSUPPORTED	WLANDRV.BIN doesn't exist.

ERA201D1_GetEtherStatistics()

Syntax

```
 EtherStatus ERA201D1_GetEtherStatistics
        (EtherDriverGetStatisticsMsg* msg)
```

Description

This gets statistics of the network interface.

Parameters

msg statistics of the network interface

Returned value

ETHER_ OK	Success
ETHER_INVALID_PORT	No WLAN card exists.
ETHER_UNSUPPORTED	WLANDRV.BIN doesn't exist.

ERA201D1_GetWLANSettings()

Syntax

EtherStatus ERA201D1_GetWLANSettings
(EtherDriverGetWLANSettingsMsg* msg)

Description

This gets settings of the wireless network.

Parameters

msg settings of the wireless network

Returned value

ETHER_ OK	Success
ETHER_INVALID_PORT	No WLAN card exists.
ETHER_UNSUPPORTED	WLANDRV.BIN doesn't exist.

ERA201D1_GetWLANStatistics()

Syntax

EtherStatus ERA201D1_GetWLANStatistics
(EtherDriverGetWLANStatisticsMsg* msg)

Description

This gets statistics for the wireless network.

Parameters

msg statistics for the wireless network

Returned value

ETHER_ OK	Success
ETHER_INVALID_PORT	No WLAN card exists.
ETHER_UNSUPPORTED	WLANDRV.BIN doesn't exist.

Changes from 20020603-E-001 to 20020730-E-002

Page 1

About Registered Trademarks

Change:

Acrobat and Adobe is a registered trademark of Adobe Systems Incorporated.

To:

Acrobat and Adobe **are** registered **trademarks** of Adobe Systems Incorporated.

Page 10

SetReadyEntry(), Description

Change:

This sets entry for when a subject receives an ASSERT-READY or DEASSERT-READYmessage. This setting is done in DoInit().

To:

This sets entry for a subject **to** receive ASSERT-READY or DEASSERT-READY messages. This setting **should be** done in DoInit().

SetReadyEntry(), Parameters

Change:

entry Entry for receiving **an** ASSERT-READY or DEASSERT-READYmessage

To:

entry Entry for receiving ASSERT-READY or DEASSERT-READY **messages**

GetID(), Description

Change:

This gets the SubjectI-D of a subject. The SubjectID is a unique value among subjects

To:

This gets the **SubjectID** of a subject. The SubjectID is a unique value among subjects

Page 11

SetBufferSize(), Description

Change:

This sets the maximum buffer size prepared for each observer in subject. This setting is done in DoInit().

To:

This sets the maximum buffer size **(number of entries)** prepared **in the subject** for each observer. This setting **should be** done in DoInit().

SetBufferSize(), Parameters

Change:

size The maximum buffer size for each observer

To:

size The maximum buffer size **(number of entries)** for each observer

GetBufferSize(), Description

Change:

This returns the buffer size that was set in DoInit().

To:

This returns the buffer size **(number of entries)** that was set in DoInit().

GetBufferSize(), Returned value

Change:

Current buffer size

To:

Current buffer size **(number of entries)**

SetNotifyUnitSize(), Description

Change:

… For example, data is composed of a header part and a body part, and both SetData() and NotifyObservers() are executed for the data. … In this case, SetData() and NotifyObserver() are called once.

To:

… For example, **some** data **may be** composed of a header part and a body part**, with each part requiring SetData(), followed by the execution of NotifyObservers()**. … In this case, SetData() and NotifyObserver() are called once **respectively for each transmission**.

Page 12

GetNotifyUnitSize(), Description

Change:

This returns the number of SetData() call to make the minimum unit of transmission data.

To:

This returns the number of SetData() **calls** to make the minimum unit of transmission data.

SetData(const void* buf, size_t size), Description

Change:

This sets data (address and size) to the buffers of all the observers. Because the specified data is copied to the shared memory segment, you can overwrite the area pointed to by 'buf' after calling this function. If overflow occurs, the oldest data waiting for transmission is overwritten by the current data. Use RemainBuffer() to check for overflow beforehand.

To:

In this function, the data region specified by 'buf' and 'size' are copied to a shared memory segment. Then, the information of the shared memory segment is set to the transmission buffers for all the observers. Because the specified region is copied to a shared memory segment, you can overwrite the source region after calling this function. If **a buffer** overflow occurs, the oldest **entry** waiting for transmission is overwritten by the current **information**. Use RemainBuffer() to check for **buffer** overflow beforehand.

SetData(const void* buf, size_t size), Parameters

Change:

size	The size of data

To:

size	The size of data **in bytes**.

SetData(const ObserverInfo& info, const void* buf, size_t size), Description

Change:

This sets data (address and size) to the buffer of the specified observer. Because this function can omit the call to FindObserver(), this function is more efficient than SetData(const ObserverID&, const void*, size_t). … If overflow occurs, the oldest data waiting for transmission is overwritten by the current data. Use RemainBuffer() to check for the overflow beforehand.

To:

In this function, the data region specified by ‘buf’ and ‘size’ are copied to a shared memory segment. Then, the information of the shared memory segment is set to the transmission buffer for the observer specified by ‘info’. Because this function can omit the call to FindObserver(), this function is more efficient than SetData(const ObserverID&, const void*, size_t). … If **a buffer** overflow occurs, the oldest **entry** waiting for transmission is overwritten by the current **information**. Use RemainBuffer() to check for the **buffer** overflow beforehand.

SetData(const ObserverInfo& info, const void* buf, size_t size), Parameters

Change:

size	The size of data.

To:

size	The size of data **in bytes**.

Page 13

SetData(const ObserverID& id, const void* buf, size_t size), Description

Change:

This function is the same as SetData(*FindObserver(id), buf, size). That is, this sets data (address and size) to the buffer of the specified observer. Because the specified data is copied to the shared memory segment in this function, you can overwrite the area pointed to by 'buf' after calling this function. If overflow occurs, the oldest data waiting for transmission is overwritten by the current data. Use RemainBuffer() to check for the overflow beforehand.

To:

This function is the same as SetData(*FindObserver(id), buf, size). **That is, the data region specified by ‘buf’ and ‘size’ are copied to a shared memory segment. Then, the information of the shared memory segment is set to the transmission buffer for the observer specified by ‘id’.** Because the specified **region** is copied to a shared memory segment, you can overwrite **the source region** after calling this function**.** If **a buffer** overflow occurs, the oldest **entry** waiting for transmission is overwritten by the current **information**. Use RemainBuffer() to check for the **buffer** overflow beforehand.

SetData(const ObserverID& id, const void* buf, size_t size), Parameters

Change:

id	The observer ID. In case the ‘id’ is invalid for subject, the result or effect of this function is undefined.

To:

id	The observer ID. In case the ‘id’ is invalid for **the present** subject, the result or effect of this function is undefined.

SetData(const ObserverID& id, const void* buf, size_t size), Parameters

Change:

size	The size of data

To:

size	The size of data **in bytes**.

SetData(RCRegion* region), Description

Change:

This sets the specified shared memory segment, region, to the buffers of all observers. If overflow occurs, the oldest data waiting for transmission is overwritten. To check for the overflow beforehand, use RemainBuffer(). …

To:

This sets the **information of the** shared memory segment **specified by** ‘region’, to the **transmission** buffers **fo**r all observers. If **a buffer** overflow occurs, the oldest **entry** waiting for transmission is overwritten. To check for the **buffer** overflow beforehand, use RemainBuffer(). …

Page 14

SetData(const ObserverInfo& info, RCRegion* region), Description

Change:

… That is, this function sets the specified shared memory segment, region, to the buffer of the specified observer. If overflow, occurs the oldest data waiting for transmission is overwritten. To check for overflow beforehand, use RemainBuffer(). …

To:

… That is, this function sets the **information of the** shared memory segment **specified by ‘region’**, to the transmission buffer for the observer **specified by ‘info’**. If a **buffer** overflow occurs, the oldest **entry** waiting for transmission is overwritten. To check for **buffer** overflow beforehand, use RemainBuffer(). …

SetData(const ObserverInfo& info, RCRegion* region), Parameters

Change:

id	The observer ID. If the 'id' is invalid for subject, the result or effect of this function is undefined.

To:

info	**The observer information. For example, the ObserverInfo type can be obtained by accessing the data that ObserverConstIterator points to, which is obtained by calling OSubject::begin().**

SetData(const ObserverID& id, RCRegion* region), Description

Change:

… That is, this sets the shared memory segment specified by argument region, to the buffer for the specified observer. In case of overflow, the oldest data for transmission is overwritten. In order to know the overflow beforehand, use RemainBuffer(). …

To:

… That is, this sets the **information of the** shared memory segment specified by argument ‘region’, to the **transmission** buffer for the observer **specified by ‘id’**. In case of **a buffer** overflow, the oldest **entry** for transmission is overwritten.   In order to know the **buffe**r overflow beforehand, use RemainBuffer(). …

SetData(const ObserverID& id, RCRegion* region), Parameters

Change:

id	… In case the 'id' is invalid for subject, the result or effect of this function is undefined.

To:

id	… In case the 'id' is invalid for **the present** subject, the result or effect of this function is undefined.

SetData(OShmPtrBase& p), Description

Change:

This sets the specified shared memory segment to the buffers of all observers. If overflow occurs, the oldest data waiting for transmission is overwritten. To check for overflow beforehand, use RemainBuffer().

To:

This sets the **information of the** shared memory segment **specified by 'p'** to the **transmission** buffers **for** all observers. If **a buffer** overflow occurs, the oldest **entry** waiting for transmission is overwritten. To check for **buffer** overflow beforehand, use RemainBuffer().

Page 15

SetData(const ObserverInfo& info, const OShmPtrBase& p), Description

Change:

This sets the specified shared memory segment to the buffer of the specified observer. … If overflow occurs, the oldest data waiting for transmission is overwritten. …

To:

This sets the **information of the** shared memory **segment specified by 'p'** to the **transmission** buffer **for** the observer **specified by 'info'**. … If **a buffer** overflow occurs, the oldest **entry** waiting for transmission is overwritten. …

SetData(const ObserverID& id, const OShmPtrBase& p), Description

Change:

This sets the specified shared memory segmentto the buffer of the specified observer. If overflow occurs, the oldest data waiting for transmission is overwritten. To check for overflow beforehand, use RemainBuffer(). …

To:

This sets the **information of the** shared memory **segment specified by 'p'** to the **transmission** buffer **for** the observer **specified by 'id'**. If **a buffer** overflow occurs, the oldest **entry** waiting for transmission is overwritten. To check for **buffer** overflow beforehand, use RemainBuffer(). …

SetData(const ObserverID& id, const OShmPtrBase& p), Parameters

Change:

id … In case the 'id' is invalid for subject, the result or effect of the function is undefined.

To:

id … In case the 'id' is invalid for **the present** subject, the result or effect of the function is undefined.

NotifyObserver(const ObserverInfo& observer), Description

Change:

This sends the data in the buffer to the specified observer. …

To:

This sends the data in the **transmission** buffer to the specified observer. …

Page 16

NotifyObserver(const ObserverID& id), Description

Change:

This sends the data in the buffer to the specified observer. … If the observer is not in the ASSERT-READY or DEASSERT-READY state, the data is kept in buffer and is sent soon after the observer's state becomes ASSERT-READY. …

To:

This sends the data in the **transmission** buffer to the specified observer. … If the observer is not in the ASSERT-READY or DEASSERT-READY state, the data is kept in **the** buffer and is sent soon after the observer's state becomes ASSERT-READY. …

NotifyObservers(void), Description

Change:

This sends the data in the buffer to all of the observers. …

To:

This sends the data in the **transmission** buffers to all of the observers. **This performs the followings for each observer**. …

RemainBuffer(const ObserverInfo& observer), Description

Change:

This returns the remaining number of buffer elements for the specified observer. If SetData() is called more than the number of times obtained by the returned value, the old data in buffer is deleted.

To:

This returns the remaining number of **transmission** buffer **entries** for the specified observer. If SetData() is called more than the number of times obtained by the returned value, the data in **the** buffer is deleted **in oldest-first manner**.

Page 17

RemainBuffer(const ObserverID& id) , Description

Change:

This returns the remaining number of buffer elements for the specified observer. If SetData() is called more than the number of times obtained by the returned value, the old data in buffer is deleted. …

To:

This returns the remaining number of **transmission** buffer elements for the specified observer. If SetData() is called more than the number of times obtained by the returned value, the data in **the** buffer is deleted **in oldest-first manner**. …

RemainBuffer(const ObserverID& id) , Returned value

Change:

Remaining number of buffer elements

To:

Remaining number of buffer elements. **0 if observer ID is invalid**.

RemainBuffer(void), Description

Change:

This returns the remaining number of buffer elements for the specified observer. If SetData() is called more than the number of times obtained by the returned value, the old data in the buffer is deleted.

To:

This returns the remaining number of **transmission** buffer elements for observers. **The number is the minimum value among the observers**. If SetData() is called more than the number of times obtained by the returned value, the data in the buffer is deleted **in oldest-first manner**.

ClearBuffer(void), Description

Change:

This clears the transmission buffers of all observers.

To:

This clears the transmission buffers **for** all observers.

Page 18

NumberOfObservers(void), Description

Change:

This returns the number of observers connecting to subject.

To:

This returns the number of observers connecting to **the present** subject.

NumberOfObservers(void), Returned value
Change:

The number of observers connecting to subject

To:

The number of observers connecting to **the present** subject

begin(), Description
Change:

This returns the iterator that points the first observer in the list of observers that connect to this subject.

To:

This returns the iterator that points **to** the first observer in the list of observers that connect to **the present** subject.

end(), Description
Change:

This returns the invalid iterator that points to the location after the last observer in the list of observers that connect to this subject.

To:

This returns the invalid iterator that points to the location after the last observer in the list of observers that connect to **the present** subject.

Page 19

FindObserver(), Description
Change:

… If the observer with id is not found, the invalid iterator is returned.

To:

… If the observer with id is not found, **an** invalid iterator is returned.

IsAllReady(void) , Returned value
Change:

Non-zero … If NotifyObserver() is executed under this state, a message is immediately sent to the observers that require the message.

To:

Non-zero … If **NotifyObservers()** is executed under this state, a message is immediately sent to the observers that require the message.

IsAnyReady(void), Returned value
Change:

Non-zero At least one observer in the ASSERT-READY state.

To:

Non-zero At least one observer **is** in the ASSERT-READY state.

Page 20

IsReady(const ObserverInfo& info), Description
Change:

This sees if the specified observer is in an ASSERT-READY state or not.

To:

This sees if the specified observer is in an ASSERT-READY state.

IsReady(const ObserverInfo& info), Parameters

Change:

info The observer information. For example, type ObserverInfo can be obtained by accessing the data that type ObserverConstIterator points, which is obtained by calling OSubject::begin().

To:

info

The observer information. For example, type ObserverInfo can be obtained by accessing the data that type ObserverConstIterator points **to**, which is obtained by calling OSubject::begin().

IsReady(const ObserverInfo& info), Returned value

Change:

Non-zero The specified observer is in an ASSERT-READY state.

To:

Non-zero The specified observer is in **the** ASSERT-READY state.

IsReady(const ObserverInfo& info), Returned value

Change:

Zero The specified observer is not in an ASSERT-READY state.

To:

Zero The specified observer is not in **the** ASSERT-READY state.

IsReady(const ObserverID& id), Returned value

Change:

Zero The specified observer is not in the ASSERT-READY state.

To:

Zero The specified observer is not in the ASSERT-READY state**, or ObserverID is invalid.**

ReadyStatus(const ObserverInfo& info), Returned value

Change:

A positive value It received an ASSERT-READY message from the observer, which is specified by subject. (ASSERT-READY state)

To:

A positive value **The subject** received an ASSERT-READY message from the **specified observer**. (ASSERT-READY state)

ReadyStatus(const ObserverInfo& info), Returned value

Change:

Zero Because the observer, which is specified by subject, has not sent any message yet, the state is unknown.

To:

Zero Because the **specified observer** has not sent **a** message yet, the state is unknown.

ReadyStatus(const ObserverInfo& info), Returned value

Change:

A negative value It received a DEASSERT-READY message from the Observer, which is specified by subject. (DEASSERT-READY state)

To:

A negative value **The subject** received a DEASSERT-READY message from the **specified observer**. (DEASSERT-READY state)

Page 21

ReadyStatus(const ObserverID& id), Returned value

Change:

A positive value — It received an ASSERT-READY message from the observer, which is specified by subject. (ASSERT-READY state)

To:

A positive value — **The subject** received an ASSERT-READY message from the **specified observer**. (ASSERT-READY state)

ReadyStatus(const ObserverID& id), Returned value

Change

Zero — Because the observer, which is specified by subject, has not sent any message yet, the state is unknown.

To:

Zero — Because the **specified** observer has not sent **a** message yet, the state is unknown. **Or, observer ID is invalid**.

ReadyStatus(const ObserverID& id), Returned value

Change:

A negative value — It received a DEASSERT-READY message from the Observer, which is specified by subject. (DEASSERT-READY state)

To:

A negative value — **The subject** received a DEASSERT-READY message from the **specified observer**. (DEASSERT-READY state)

ControlHandler(), Description

Change:

This sets up a subject in accordance with the received OControlHandler. This is called during the connection phase of objects.

To:

This sets up a subject in accordance with the received **OControlMessage**. This is called during the connection phase of objects.

Page 22

SenderID(), Description

Change:

This returns the observer ID of the observer that sends OReadyEvent.

To:

This returns the observer ID of the observer that **has sent** OReadyEvent.

Page 23

SetNotifyEntry(), Description

Change:

This sets the entry for when the observer receives a NOTIFY message. This setting is done in DoInit().

To:

This sets the entry for the observer **to receive** NOTIFY **messages**. This setting **should be** done in DoInit().

Page 24

SetBufCtrlParam(), Description

Change:

This sets the necessary control parameters of the buffer for observers of a subject. This setting is done in DoInit().

To:

This sets the necessary control parameters of the buffers **that the subject holds** for observers. This setting **should be** done in DoInit().

SetBufCtrlParam(), Parameters

Change:

skip This specifies the data to skip (a sampling interval) to reduce the amount of received data. The default value is zero, which means no sampling.

To:

skip This specifies the data-skip (a sampling interval) to reduce the amount of **receiving** data. The default value is zero, which means no **sub-**sampling.

SetBufCtrlParam(), Parameters

Change:

min This specifies the minimum amount of data when a subject sends the NOTIFY message to an observer. The default value is one. If you adequately set this parameter, you can reduce the amount of received data without data loss.

To:

min This specifies the minimum amount of data **units** when a subject sends the NOTIFY message to an observer. The default value is one. If you adequately set this parameter, you can reduce the **frequency of data-receiving** without data loss.

SetBufCtrlParam(), Parameters

Change:

max By an observer becomes ASSET-READY state, this specifies the maximum amount of a transmission buffer size that a subject should hold. This parameter must be greater than or equal to 'min'. The default value is one. Only the transmission data is held in the buffer when the value is one.

To:

max This specifies the maximum transmission buffer size **(units)** that a subject should hold **until an observer's state becomes ASSET-READY**. This parameter must be greater than or equal to 'min'. The default value is one. Only the **last** transmission data **unit** is held in the buffer when the value is one.

SetSkip(), Description

Change:

This sets the necessary control parameter of the buffer for observers of a subject. This setting is done in DoInit(). This function is available to keep compatibility with previous software. This function is the same as SetBufCtrParam(skip, 1, 1).

To:

This sets the necessary control parameter of the **buffers that the subject holds** for observers. This setting **should be** done in DoInit(). This function is available to keep compatibility with previous software. This function is the same as SetBufCtrlParam(skip, 1, 1).

SetSkip(), Parameters

Change:

skip This specifies the data to skip (the sampling interval) to reduce the amount of received data. The default value is zero, which is no sampling.

To:

skip This specifies the data-skip (the sampling interval) to reduce the amount of **receiving** data. The default value is zero, which **means** no **sub-**sampling.

Page 26

NumberOfSubjects(), Description

Change:

This returns the number of subjects connecting to an observer.

To:

This returns the number of subjects connecting to **the present** observer.

NumberOfSubjects(), Returned value

Change:

The number of subjects connecting to an observer

To:

The number of subjects connecting to **the present** observer

begin(), Description

Change:

This returns the iterator that points the first subject in the subject list that connects to an observer.

To:

This returns the iterator that points **to** the first subject in the subject list that connects to **the present** observer

begin(), Returned value

Change:

The iterator that points the first subject

To:

The iterator that points **to** the first subject

end(), Description

Change:

This returns the invalid iterator that points to the location after the last subject in the subject list that connects to an observer.

To:

This returns the invalid iterator that points to the location after the last subject in the subject list that connects to **the present** observer.

Page 27

ConnectHandler(), Parameters

Change:

status This indicates the status of the function for any user-defined initialization/resource allocation. The default value is oSUCCESS, and in case it is not oSUCCESS, **a** connection will be refused.

To:

status This indicates the status of the function for any user-defined initialization/resource allocation. The default value is oSUCCESS, and in case it is not oSUCCESS, connection will be refused.

Page 28

title of chapter

Change:

2.4 NotifyEvent class

To:

2.4 **ONotifyEvent** class

NumOfNotify(void), Description.

Change:

This returns the number of times that ONotifyEvent() was executed to send data.

To:

This returns the number of times that ONotifyEvent() was executed **for the** data **that has been sent**.

Page 30

Line 2

Change:

This class has a pointer to the shared memory segment and controls the reference counter for the memory segment. The following are member functions.

To:

This class has a pointer to the shared memory segment and controls the reference counter for the memory segment. The following are member functions. **You cannot instantiate this class on the local stack**.

RCRegion(void), Description

Change:

This is constructor. It constructs the instance pointing NULL.

To:

This is constructor. It constructs the instance pointing **to** NULL.

RCRegion(size_t size), Description

Change:

This reserves the shared memory segment with the specified size, and constructs an instance pointing to this memory segment.

To:

This reserves **a** shared memory segment with the specified size, and constructs an instance pointing to this memory segment.

RCRegion(MemoryRegionID memID, size_t offset, void* baseAddr=NULL, size_t size=0), Parameters

Change:

size	Data size

To:

size	Data size **in bytes**

Page 31

~RCRegion(), Description

Change:

… Instead of calling the destructor, you can call RemoveReference().

To:

… Instead of calling the destructor, **you must** call **RCRegion::**RemoveReference().

Page 32

Size(), Returned value

Change:

The size of data on the shared memory segment.

To:

The size **(in bytes)** of data **in** the shared memory segment.

Page 33

SetSize(), Description

Change:

This sets size to the value returned by RCRegion::Size(). This function is used so the user can optimization memory allocation routines.

To:

This sets the value returned by RCRegion::Size() **to 'size'**. This function is used so the user can **apply** optimization **in original** memory allocation routines.

ReserveSharedMemory(), Description

Change:

…In case enough shared memory segments do not exist when a function is called, the necessary memory segment will be allocated…

To:

…In case enough shared memory segments do not exist when **this** function is called, the necessary memory segment will be allocated…

ReserveSharedMemory(), Parameters

Change:

size When SetData(ptr, size) is called, a memory segment is used. The size for the memory segment is reserved here.

To:

size **The size of the memory segment to be reserved, for future SetData(ptr, size) calls.**

Page 34

OShmPtrBase(const OShmPtrBase& p), Description

Change:

This constructs OShmPtrBase that refers to the same region as the specified OShmPtrBase refers.

To:

This constructs OShmPtrBase that refers to the same region as the specified OShmPtrBase refers **to**.

Page 35

Base(), Description

Change:

This returns the base address of data of a shared memory segment.

To:

This returns the base address of data **in** a shared memory segment.

Base(), Returned value

Change:

The base address of data of a shared memory segment

To:

The base address of data **in** a shared memory segment

Size(), Description

Change:

This returns the size of data of a shared memory segment.

To:

This returns the size of data **in** a shared memory segment.

Size(), Returned value

Change:

The size of data of a shared memory segment

To:

The size of data **in** a shared memory segment

Page 36

MemID(), Returned value

Change:

Shared memory ID

To:

ID of a shared memory segment

Offset(), Description

Change:

This returns the offset of a shared memory segment. …

To:

This returns the offset **to the** data segment. …

Offset(), Returned value

Change:

The offset of a data segment

To:

The offset **to the** data segment

Page 38

OShmPtr(size_t n), Description

Change:

This reserves a shared memory segment with sizeof(T)*n size, and constructs an array of OShmPtr<T> with n element. This function internally calls Allocate(n). A constructor Type T is not called.

To:

This reserves a shared memory segment with **the size of** sizeof(T)*n, and constructs an array of OShmPtr<T> with n elements. This function internally calls Allocate(n). A constructor **for type** T is not called.

OShmPtr(size_t n), Parameters

Change:

n	An array of OShmPtr<T> with n element

To:

n	An array of OShmPtr<T> with n **elements**

Allocate(), Description

Change:

This reserves a shared memory segment with the size of sizeof(T)*n, and allocates an array with type T and n elements. The reference counter controls this newly constructed shared memory segment. A constructor with type T is not called.

To:

This reserves a shared memory segment with the size of sizeof(T)*n, and allocates an array **of** type T **with** n elements. The reference counter controls this newly constructed shared memory segment. A constructor **for** type T is not called.

Allocate(), Parameters

Change:

n The number of elements of an array with type T

To:

n The number of elements of an array **of** type T

Page 39

OShmPtr<T>::Proxy operator*(void), Returned value

Change:

First element The first element in the array

To:

The first element in the array

Page 41

OVirtualRobotComm.Effector.OCommandVectorData.O

Change:

… OCommnadVectorData reserves a shared memory with OPEN-R::NewCommandVectoreData(). …

To:

… **You can reserve** a shared memory **for OCommandVectorDat**a with OPEN-R::NewCommand**Vector**Data(). …

OVirtualRobotComm.FbkImageSensor.OFbkImageVectorData.S

Change:

… Three sheets of YCrCb and a sheet of CDT are sent to access the image data.

To:

… The sending data structure is OFbkImageVectorData. Three sheets of YCrCb and a sheet of CDT are **included in** the image data.

Page 42

OVirtualRobotAudioComm.Speaker.OSoundVectorData.O

Change:

… OSoundVectorData reserves a shared memory with OPENR::NewSoundVectorData(). …

To:

… You can reserve a shared memory **for OSoundVectorData** with OPENR::NewSoundVectorData(). …

Page 43

ODataVectorInfo, Members

Change:

memRegionID This is the ID of a shared memory segment that has data.

To:

memRegionID This is the ID of a shared memory segment that **holds** data.

ODataVectorInfo, Members

Change:

TotalSize This is the size of a shared memory that has data.

To:

TotalSize This is the size of a shared memory that **holds** data.

Page 44

ODataVectorInfo, Members

Change:

wait This waits for commands and the output of sound, for the number of frames (8msec) specified by ‘wait’.

To:

wait **Delays** commands and the output of sound, for the number of frames (**in units of** 8msec) specified by ‘wait’.

ODataVectorInfo, Members

Change:

optional[odataOPTIONAL_MAX]

It is used for the delivery of the information between the object that sends OSensorFrameVectorData and the object that sends OCommandVectorData, OSoundVectorData. The data which is specified with optOffset and optSize is updated, and the data is copied to optional[] of OSoundFrameVectorData.

To:

optional[odataOPTIONAL_MAX]

It is used for the delivery of the information between the object that **receives** OSensorFrameVectorData and the object that sends OCommandVectorData, OSoundVectorData. The data **in optional[] (whose range** is specified with optOffset and optSize**)** is updated, and the data is copied to optional[] of **OSensorFrameVectorData**.

Page 45

4.2.1 OcommandVectorData, Description

Change:

This is a data structure that has joint and LED commands…..
It is possible to keep different kinds of commands with one OCommanVectorData.

To:

This is a data structure that **holds** joint and LED commands.….
It is possible to keep different kinds of commands **in** one OCommandVectorData.

Page 46

OCommandInfo, Members

Change:

frameNumber The frame number of the first frame that is processed by the command.

To:

frameNumber The frame **sequence** number **when** the first frame is processed by the command **will be stored here.**

OCommandInfo, Members

Change:

numFrames … The value of numFrames (maximum of MAX_FRAMES=16) is the number of commands that are processed.

To:

numFrames … **Only** numFrames **frames** out of **ocommand**MAX_FRAMES(=16) are processed.

OCommandInfo, Members

Change:

frameSize This is the size (8 bytes) of command data in one framethat OCommandData keeps.

To:

frameSize This is the size (8 bytes) of command data in one **frame that** OCommandData keeps.

OCommandInfo, Members

Change:

dataOffset This is an offset of OCommadData corresponding to OCommadInfo. …

To:

dataOffset This is an offset **to** OComma**n**dData corresponding to OComma**n**dInfo. …

OCommandInfo, Members

Change:

dataSize	This is the data size (128 bytes) of OCommadData corresponding to OCommandInfo.

To:

dataSize	This is the data size (128 bytes) of OComma**n**dData corresponding to OCommandInfo.

Page 47

OCommandData, Members

Change:

value[ocommandMAX_FRAMES]

This is command data. OCommandData can have data for a maximum of ocommandMAX_FRAMES (=16) frames. The valid number of frames is specified by numFrames of OCommandInfo.

To:

value[ocommandMAX_FRAMES]

This is command data. OCommandData can **hold** data for a maximum of ocommandMAX_FRAMES (=16) frames. The number of **valid** frames is specified by numFrames of OCommandInfo.

OJointCommandValue2, Members

Change:

value This is a value to be set to a joint. The unit is micro radians (10-6 rad). In the case of 360 deg, the value would be 3141592.

To:

value This is a value to be set to a joint. The unit is micro radians (10-6 rad). In the case of **180** deg, the value would be 3141592.

Page 48

OJointCommandValue3, Members

Change:

value It is a value to be set to a plunger. value can be ojoint3-STATE0 or ojoint3-STATE1.

To:

value It is a value to be set to a plunger. value can be **ojoint3_STATE0** or **ojoint3_STATE1**.

OLEDCommandValue2, Description

Change:

…The control of an LED is specified by ON/OFF and its time. …

To:

…The control of an LED is specified by ON/OFF and its **duration. …**

OLEDCommandValue2, Members

Change:

period … The shortest length of time is 8ms.

To:

period … The **unit** of time is 8ms.

Page 50

OSensorFrameInfo, Description

Change:

This has the type of element of OSensorFrameVectorData, the number of frames in sensor data and the offset to sensor data.

To:

This **contains** the type of element of OSensorFrameVectorData, **OPrimitiveID**, the number of frames in sensor data and the offset to sensor data.

OSensorFrameInfo, Members

Change:

frameNumber This is the frame number when the first data of a corresponding OSensorFrameData is obtained.

To:

frameNumber This is the frame **sequence** number when the first data of a corresponding OSensorFrameData is obtained.

OSensorFrameInfo, Members

Change:

dataOffset This is the offset of OSensorFrameData corresponding to OSensorFrameInfo …

To:

dataOffset This is the offset **to** OSensorFrameData corresponding to OSensorFrameInfo …

OSensorFrameInfo, Members

Change:

dataSize This is a data size (128 bytes) of OSensorFrameData corresponding to OSensorFrameInfo.

To:

dataSize This is a data size (**256** bytes) of OSensorFrameData corresponding to OSensorFrameInfo.

Page 51

OSensorFrameData, Members

Change:

frame[osensorframeMAX_FRAMES]

This is sensor data. OSensorFrameData can have data for themaximum number of frames (osensorframeMAX_Frames =16). The valid number of frames is specified by numFrames in SensorFrameinfo.

To:

frame[osensorframeMAX_FRAMES]

This is sensor data. OSensorFrameData can have data for **the maximum** number of **osensorframeMAX_Frames (=16) frames**. The number of **valid** frames is specified by numFrames in OSensorFrameinfo.

Page 54

OSwitchStatus, Members

Change:

value This is the status of a switch. It is either oswitchON or oswitchOFF.

To:

value This is the status of a switch, **converted from an A/D signal value obtained from a switch**. It is either oswitchON or oswitchOFF.

Page 56

OFbkImageInfo, Description

Change:

… This is the data structure that has a YCrCb image and a CDT image.

To:

… This is the **image information**. This is the data structure that **holds** a YCrCb image and a CDT image.

OFbkImageInfo, Members

Change:

frameNumber	This is the frame number of the image when it was obtained.

To:

frameNumbe	This is the frame **sequence** number when **the image** was obtained.

Page 57

OFbkImage, Syntax

Change:

OFbkImage(OFbkImageInfo* info, **byte*,** byte* data, OFbkImageBand band)

To:

OFbkImage(OFbkImageInfo* info, byte* data, OFbkImageBand band)

OFbkImage, Description

Change:

…ofbkimageBAND_Y, ofbkimageBAND_Cr, ofbkimageBAND_Cb...

To:

…ofbkimageBAND_Y, ofbkimageBAND_Cr, ofbkimageBAND_Cb **for band. …**

Page 59

Pixel(), Description

Change:

This returns the pixel values of an image with coordinate (x, y). …

To:

This returns the pixel **value** of an image with coordinate (x, y). …

Pixel(), Returned value

Change:

The pixel values of an image with coordinate (x, y)…

To:

The pixel **value** of an image with coordinate (x, y)…

Page 60

4.3 Communication with OVirtualRobotAudioComm

Change:

The following is the data for communication with OirtualRobotAudioComm.

OSoundVectorData	A sound data

The data is created in a shared memory segment. The contents of this data are placed in the following order: ODataVectorInfo of a common header, the array of the information block of each element, and the array of the data.

To:

The following is the data for communication with O**V**irtualRobotAudioComm.

OSoundVectorData	**Sound** data

The data is created in a shared memory segment. The contents of this data are placed in the following order: ODataVectorInfo **as** a common header, the array of the information block **about** each element, and the array of the data body.

4.3.1 OSoundVectorData, Description

Change:

This is the data structure that has sound data. …

To:

This is the data structure that **holds** sound data. …

OSoundInfo, Description

Change:

This is the data structure that has sound data information.

To:

This is the data structure that **holds** sound data information.

Page 61

OSoundInfo, Members

Change:

frameNumber For the output of sound, frameNumber is the frame number when OVirtualRobot processes the first frame of sound. For input of sound, the frame number when data was input is used.

To:

frameNumber For the output of sound, frameNumber is the frame **sequence** number when OVirtualRobot processes the first frame of sound. For input of sound, the frame **sequence** number when data was input is used.

Page 62

4.4.1 OCdtVectorData, Description

Change:

This is a data structure that has a color detection table. It can have a maximum table of ocdNUM_CHANNELS (=8). …

To:

This is a data structure that **holds** a color detection table. It can have a maximum of ocdNUM_CHANNELS (=8) **tables**. …

OCdtInfo, Description

Change:

… The values of Cr and Cb are **an** offset binary ranging between 0x0 and 0xff.

To:

… The values of Cr and Cb are offset binary ranging **from** 0x0 **to** 0xff.

Page 63

OCdtInfo, Members

Change:

type This is the data type. ofodataCDT is used.

To:

type This is the data type. **odataCDT** is used.

OCdtInfo, Members

Change:

primitiveID The PrimitiveID of OFbkImageSensor that sets the CDT.

To:

pr**i**mitiveID The PrimitiveID of OFbkImageSensor that the CDT **is set to**.

OCdtInfo, Members

Change:

channel This is a channel of the CDT that sets a table.

To:

channel This is a channel of the CDT that a table **is set to**.

Page 66

OPENR::NewCommandVectorData(), Parameters

Change:

numCommands The number of element in OCommandData

To:

numCommands The number of **elements** in OCommandData

OPENR::NewCommandVectorData(), Parameters

Change:

memID MemoryRegionID of the shared memory in OCommandVectorData

To:

memID MemoryRegionID of the shared memory **for** OCommandVectorData

OPENR::DeleteCommandVectorData(), Description

Change:

This releases the shared memory in OCommandVectorData.

To:

This releases the shared memory **for** OCommandVectorData.

OPENR::DeleteCommandVectorData(), Parameters

Change:

memID MemoryRegionID of the shared memory in OCommandVectorData

To:

memID MemoryRegionID of the shared memory **for** OCommandVectorData

Page 67

OPENR::NewSoundVectorData(), Description

Change:

This reserves shared memory for OSoundVectorData. Set the valid number of elements with SetNumData().

To:

This reserves shared memory for OSoundVectorData. **vectorInfo.numData is initialized to 0.** Set the valid number of elements with SetNumData().

OPENR::NewSoundVectorData(), Parameters

Change:

numSounds The number of element in sound data

To:

numSounds The number of **elements** in sound data

OPENR::NewSoundVectorData(), Parameters

Change:

memID MemoryRegionID of the shared memory in OSoundVectorData

To:

memID MemoryRegionID of the shared memory **for** OSoundVectorData

OPENR::DeleteSoundVectorData(), Description

Change:

This releases the shared memory in OSoundVectorData.

To:

This releases the shared memory **for** OSoundVectorData.

OPENR::DeleteSoundVectorData(), Parameters

Change:

memID MemoryRegionID of the shared memory in OSoundVectorData

To:

memID MemoryRegionID of the shared memory **for** OSoundVectorData

OPENR::NewCdtVectorData(), Parameters

Change:

memID MemoryRegionID of the shared memory in OCdtVectorData

To:

memID MemoryRegionID of the shared memory **for** OCdtVectorData

Page 68

OPENR::DeleteCdtVectorData(), Description

Change:

This releases the shared memory in OCdtVectorData.

To:

This releases the shared memory **for** OCdtVectorData.

OPENR::DeleteCdtVectorData(), Parameters

Change:

memID MemoryRegionID of the shared memory in OCdtVectorData.

To:

memID MemoryRegionID of the shared memory **for** OCdtVectorData.

OPENR::SetCdtVectorData(), Parameters

Change:

memID MemoryRegionID of the shared memory in OCdtVectorData.

To:

memID MemoryRegionID of the shared memory **for** OCdtVectorData.

Page 69

OPENR::EnableJointGain(),Parameters

Change:

primitiveID OPrimitiveID of a Joint or oprimitiveID_UDEF

To:

primitiveID OPrimitiveID of a Joint or oprimitiveID_**UNDEF**

OPENR::DisableJointGain(), Parameters

Change:

primitiveID OPrimitiveID of a joint or oprimitiveID_UDEF

To:

primitiveID OPrimitiveID of a joint or oprimitiveID_**UNDEF**

Page 70

OPENR::SetJointGain(), Parameters

Change:

primitiveID OprimitiveID of a joint or oprimitiveID_UDEF

To:

primitiveID OprimitiveID of a joint or oprimitiveID_**UNDEF**

OPENR::RegisterDefaultJointGain(), Description

Change:

This registers the default gain to a joint. If oprimitiveID_UNDEF is specified to primitiveID, it registers all joints opened by OPENR::OpenPrimitive() to the default gain.

To:

This registers the default gain to a joint. If oprimitiveID_UNDEF is specified to primitiveID, it registers **the default gain to** all joints opened by OPENR::OpenPrimitive().

OPENR::RegisterDefaultJointGain(), Parameters

Change:

primitiveID OprimitiveID of a joint or oprimitiveID_UDEF

To:

primitiveID OprimitiveID of a joint or oprimitiveID_**UNDEF**

Page 71

OPENR::SetDefaultJointGain(), Description

Change:

… oSUCCESS is returned when the gain of a joint has succeeded.

To:

… oSUCCESS is returned when the gain of a joint has successfully been set.

OPENR::SetDefaultJointGain(), Parameters

Change:

primitiveID OPrimitiveID of the joint or oprimitiveID_UDEF

To:

primitiveID OPrimitiveID of the joint or oprimitiveID_**UNDEF**

Page 72

OPENR::NewSyncKey(), Description

Change:

… When you have exceeded 8, an oversynckeyUNDEF is substituted for the synchronization key, and oNO_SYNC_KEY is returned.

To:

… When you have exceeded 8, an **ovrsynckey**UNDEF is substituted for the synchronization key, and oNO_SYNC_KEY is returned.

Page 73

OPENR::SetMotorPower(), Description

Change:

This controls the power to motors. opowerOFF or opowerON is specified to power

To:

This controls the power to motors. opowerOFF or opowerON is specified to **‘power’**.

OPENR::GetBootCondition(), Description

Change:

… The boot condition is saved to bitmap. bootTime, batteryCapacityLow, andvibrationLevel are 0.

To:

… The boot condition is saved to bitmap. bootTime, **bootTimeType**, and vibrationLevel are **invalid**.

OPENR::GetBootCondition(), Description

Change:

Types of boot conditions

obcbBOOT_TIMER =0x0001 Starts with time.

To:

Types of boot conditions

obcbBOOT_TIMER =0x0001 Starts **on scheduled time**.

Page 74

OPENR::GetPowerStatus(), Description

Change:

This gets the power status, which is defined by the following structure.

To:

This gets the **hardware** status, which is defined by the following structure.

OPENR::GetPowerStatus(), Description

Change:

The following are the units for each member.

remainingCapacity

The battery remaining capacity (1%bit, 0 - 100%)

temperature

The battery temperature (0.1Kelvin/bit, 0 - 500.0Kelvin)

…

voltage

The battery voltage (1mV/bit, 0 - 65535mV)

current

The battery current (1mA/bit, -32768 – 32767mA)

To:

The following are the units for each member.

remainingCapacity

The battery remaining capacity **(%**, 0 - 100%)

temperature

The battery temperature **(0.1Kelvin**, 0 - 500.0Kelvin)

…

voltage

The battery voltage (**mV**, 0 - 65535mV)

current

The battery current (**mA**, –32768 - 32767mA)

OPENR::GetPowerStatus(), Description

The following are added.

robotStatus Indicates general hardware status.

orsbPAUSE = 0x00000001
Pause switch is on.

orsbMOTOR_POWER = 0x00000002
Motor power is on.

orsbVIBRATION_DETECT = 0x00000004
Vibration detected.

orsbEX_PORT_CONNECTED = 0x00000008
Connected to an external connector. External connectors include connectors of the AC adaptor and the station.

orsbSTATION_CONNECTED = 0x00000010
Connected to the station.

orsbEX_POWER_CONNECTED = 0x00000020
Connected to an external power supply.

orsbBATTERY_CONNECTED = 0x00000040
Battery is connected.

orsbBATTERY_CHARGING = 0x00000080
Battery is charging.

orsbBATTERY_CAPACITY_FULL = 0x00000100
Battery capacity full.

orsbBATTERY_CAPACITY_LOW = 0x00000200
Battery capacity low.

orsbBATTERY_OVER_CURRENT = 0x00000400
Battery current too high

orsbBATTERY_OVER_TEMP_DISCHARGING = 0x00000800
Battery temperature on discharging is too high

orsbBATTERY_OVER_TEMP_CHARGING = 0x00001000
Battery temperature on charging is too high

orsbBATTERY_ERROR_OF_CHARGING = 0x00002000
Error on battery charging

orsbERROR_OF_PLUNGER = 0x00004000
Error on plunger. Unable to lock battery.

orsbOPEN_R_POWER_GOOD = 0x00008000
Power supplied to OPEN-R Bus system (3.3V)

orsbERROR_OF_FAN = 0x00010000
Error on cooling fan.

orsbDATA_STREAM_FROM_STATION = 0x00020000
The station has written data onto the datastream region.

orsbREGISTER_UPDATED_BY_STATION= 0x00040000
The station has updated some of the register region.

orsbRTC_ERROR = 0x00080000
Error on RTC (Real Time Clock)

orsbRTC_OVERFLOW = 0x00100000
Overflow occurred in RTC. (Note 1)

orsbRTC_RESET = 0x00200000
Indicates RTC has been reset. (Note 2)

orsbRTC_SET = 0x00400000
Indicates time-setting to RTC has been performed. This flag will be cleared on the notification to the entry that is monitoring this flag.

orsbSPECIAL_MODE = 0x00800000
Required to enter special mode.

orsbBMN_DEBUG_MODE = 0x01000000
Indicates BMN microcontroller is in the debug mode.

orsbCHARGER_STATUS = 0x02000000
Indicates the charging circuit in AIBO is on.

orsbPLUNGER = 0x04000000
Indicates the plunger is locked.

orsbSUSPENDED = 0x08000000

reserved

orsbSPECIAL_DATA_READ_REQ = 0x10000000

reserved

Note 1

The time is represented by the number of seconds elapsed since 2000/1/1 0:00. The data length is 32-bits (signed). Therefore, if the value exceeds 0x7fffffff, the elapsed seconds will be negative and unable to represent the time properly. Starting from year 2000, it is possible to represent time until around year 2068. This flag will be cleared when the time is set, by using the LCD panel on AIBO, via a command by the CPU, or via the station.

Note 2

If it is not charged for a long period, the local power of the RTC will be exhausted and the time kept in the RTC will be lost. This flag will also be cleared when the time is set, using the mothods described above.

batteryStatus Indicates battery status.

obsbERROR_CODE_MASK = 0x000F
Error code returned by the battery.
obsbFULLY_DISCHARGED = 0x0010
Indicates the battery is fully discharged.
obsbFULLY_CHARGED = 0x0020
Indicates the battery is fully charged.
obsbDISCHARGING = 0x0040
Indicates the battery is discharging.
obsbINITIALIZED = 0x0080
Always one
obsbREMAINING_TIME_ALARM = 0x0100
Indicates the operable battery time is short.
obsbREMAINING_CAPACITY_ALARM = 0x0200
Indicates remaining capacity of the battery is low. This is different from orsbBATTERY_CAPACITY_LOW in robotStatus.
obsbRESERVED0 = 0x0400
reserved
obsbTERMINATED_DISCHARGING_ALARM = 0x0800
Indicates discharging is terminated.
obsbOVER_TEMP_ALARM = 0x1000
Temperature is too high.
obsbRESERVED1 = 0x2000
reserved
obsbTERMINATED_CHARGING_ALARM = 0x4000
Indicates that the battery charging is terminated.
obsbOVER_CHARGED_ALARM = 0x8000
Alarm for excesive charging

Page 75

OPENR::ObservePowerStatus(), Description

Change:

When a parameter specified by notifyStatus is changed, the content that was changed is notified to entry. NotifyStatus cannot monitor fullyChargedCapacity, voltage, or current. In robotStatus and batteryStatus, when a specified bit is changed, the bit is notified.In remainingCapacity, temperature, timeDif, and volume, the following symbolic constants are defined in OPower.h. In opso*_NOTIFY_EVERY_CHANGE, when it is changed, it is notified. In opso*_NOT_NOTIFY, when it is changed, it is not notified. Excluding the above values, when a value becomes the specified value, a notification occurs. The notified message structure is OPowerStatusMessage.

…

Once ObservePowerStatus() is executed, the specified entry will be notified when the power status matches the specified notifyStatus. This occurs until OPENR::UnobservePowerStatus() is executed. Each bit roboStatus and batteryStatus of notifyStatus is notified on both rising and falling edges. remainingCapacity, temperature, timeDif, and volume are notified when each value is changed, or it becomes the specified value. When a value is specified, a notification occurs when the value becomes the specified value. However, a notification does not occur when the value is changed. After that, a notification does not occur if the value is changed from the specified value.

To:

When a parameter specified by notifyStatus is changed, **the specified** ‘**entry’ will be notified of the change**. **In** NotifyStatus, fullyChargedCapacity, **‘voltage’, or ‘current’ cannot be monitored for their changes**. **For** robotStatus and batteryStatus, **a notification will occur** when a specified bit is changed. **For** remainingCapacity, temperature, timeDif, and volume, the following symbolic constants are defined in OPower.h. **Specifying** opso*_NOTIFY_EVERY_CHANGE **for a parameter indicates notification of changes of this parameter. Specifying** opso*_NOT_NOTIFY **for a parameter indicates not to notify when parameter is changed. A value excluding** the above **two indicates notification** when **the parameter’s value** becomes the specified value. The notified message structure is OPowerStatusMessage.

…

Once ObservePowerStatus() is executed, the specified entry will be notified **every time** the power status matches the specified notifyStatus. This **continues** until OPENR::UnobservePowerStatus() is executed. **For each** bit **of robotStatus** and batteryStatus **in** notifyStatus, **a notification will occur** on both rising and falling edges. **For** remainingCapacity, temperature, timeDif, and volume, **a notification will occur** when each **parameter’s** value is changed, or it becomes the specified value. When a value is specified, a notification occurs when the parameter’s value becomes the specified value. However, a notification **will** not occur if the **parameter’s** value is changed from the specified value, **nor if the parameter’s value is unchanged**.

Page 76

OPENR::FindDesignData(), Description

Change:

… If it is found, the design data file is written to shared memory, and the starting address and ODesignDataID are returned. If you specify the reserved keyword 'SYS_CPUINFO' to a parameter, the operating frequency of the CPU and the starting address of OCPUInfo is returned. Though the keyword 'SYS_CPUINFO' is not registered to DESIGNDB.CFG, this keyword works.

To:

… If it is found, the design data file is **copied** to shared memory, and the starting address and ODesignDataID are returned. If you specify the reserved keyword 'SYS_CPUINFO' to a parameter, **you can obtain** the operating frequency of the CPU, **as** the starting address of OCPUInfo **is** returned. **Even if** the keyword 'SYS_CPUINFO' is not registered to DESIGNDB.CFG, this keyword works.

OPENR::FindDesignData(), Parameters

Change:

size	Size of design data

To:

size	Size of design data **in bytes**

OPENR::FindDesignData(), Returned value

Change:

oNOT_FOUND	The keyword or design data does not exist.

To:

oNOT_FOUND	The keyword or design data **body** does not exist.

OPENR::FindDesignData(), Returned value

Change:

oDESIGNDATA_SIZE_ZERO	The file size in design data is 0.

To:

oDESIGNDATA_SIZE_ZERO	The file size **for** design data is 0.

Page 77

OPENR::DeleteDesignData(), Description

Change:

This releases the memory in design data.

To:

This releases the memory **for** design data.

OPENR::GetRobotDesign(), Description

Change:

This gets the robot design.

To:

This gets the **'robot design'**.

OPENR::GetRobotDesign(), Parameters

Change:

robotDesign	robotDesign Robot design (ex. ERS-210)

To:

robotDesign	**'Robot design' string** (ex. **'ERS-210'**)

OPENR::GetMemoryStickStatus(), Description

Change:

This checks the status of the Memory Stick

omemorystickNOT_EXIST

No Memory Stick exists.

To:

This checks the status of **the AIBO Programming Memory Stick**

omemorystickNOT_EXIST

No **AIBO Programming Memory Stick** exists.

OPENR::GetMemoryStickStatus(), Parameters

Change:

status	The status of the Memory Stick

To:

status	The status of the **AIBO Programming Memory Stick**

Page 78

OPENR::Fatal(), Description

Change:

This sounds a warning sound with the buzzer in the microcomputer, and turns off power. Specify the kind of warning sound with fatal.

To:

This sounds a warning sound with the buzzer in the **BMN microcontroller**, and turns off power. Specify the kind of warning sound with ‘fatal’.

OPENR::Fatal(), Parameters

Change:

fatal	The kind of warning sound. **Only ofatalMEMORY_STICK is supported.**

To:

fatal	The kind of warning sound.

OPENR::Fatal(), Parameters

Change:

ofatalMEMORY_STICK	Memory Stick destruction error sound

To:

ofatalMEMORY_STICK	**AIBO Programming Memory Stick** destruction error sound

OPENR::SetTime(), Description

Change:

This sets the time of the RTC to the time specified by time. When time is a value from –12 to +12 and the time difference, that is different from the current time difference, is set, the time difference is also set to the BMN microcomputer.

To:

This sets the time specified by ‘time’ **to the time of the RTC**. **If the time difference is set in ‘time’ as** a value from –12 to +12 that is different from the current time difference, the time difference is also set to the BMN **microcontroller**.

OPENR::SetTime(), Parameters

Change:

time	Time and a time difference

To:

time	**The structure of time** and a time difference

OPENR::GetTime(), Description

Change:

This gets the time and a time difference.

To:

This gets the time and **the** time difference.

OPENR::GetTime(), Parameters

Change:

time The structure of time difference and time

To:

time The structure of **time and time difference**

Model Information ERS-210

Changes form 20020730-E-002 to 20030201-E-003

Page 18

The following are added.

2.4 Relations between the polarity of PWM and the polarity of rotation angle of joints

In OPEN-R SDK 1.1.3 r1, rotation angle of some of the joints had opposite polarity to the corresponding PWM duty. In OPEN-R SDK 1.1.3 r2, polarities of rotation angle and PWM duty are aligned for all of the joints.

Polarity of rotation angle of joint to the positive direction of PWM

	(The version of OPEN-R SDK 1.1.3)	r1	r2
PRM : /r1/c1-Joint2:j1	Neck tilt	-	+
PRM : /r1/c1/c2-Joint2:j2	Neck pan	-	+
PRM : /r1/c1/c2/c3-Joint2:j3	Neck roll	-	+
PRM : /r1/c1/c2/c3 /c4-Joint2:j4	Mouth	+	+
PRM : /r2/c1-Joint2:j1	Left fore Leg, J1 joint	+	+
PRM : /r2/c1/c2-Joint2:j2	Left fore Leg, J2 joint	-	+
PRM : /r2/c1/c2/c3-Joint2:j3	Left fore Leg, J3 joint	+	+
PRM : /r3/c1-Joint2:j1	Left hind leg, J1 joint	-	+
PRM : /r3/c1/c2-Joint2:j2	Left hind leg, J2 joint	-	+
PRM : /r3/c1/c2/c3-Joint2:j3	Left hind leg, J3 joint	+	+
PRM : /r4/c1-Joint2:j1	Right fore leg, J1 joint	-	+
PRM : /r4/c1/c2-Joint2:j2	Right fore leg, J2 joint	-	+
PRM : /r4/c1/c2/c3-Joint2:j3	Right fore leg, J3 joint	+	+
PRM : /r5/c1-Joint2:j1	Right hind leg, J1 joint	+	+
PRM : /r5/c1/c2-Joint2:j2	Right hind leg, J2 joint	-	+
PRM : /r5/c1/c2/c3-Joint2:j3	Right hind leg, J3 joint	+	+
PRM : /r6/c1-Joint2:j1	Tail pan	-	+
PRM : /r6/c2-Joint2:j2	Tail tilt	-	+

Page 19

3.2 Speaker

The following are added.

ospksndMONO16K16B

Changes from 20020603-E-001 to 20020730-E-002

Page 7

1.2.2 Leg

Change:

3DOF (Waist:1DOF, Shoulder:1DOF, Knee:1DOf) x2
3DOF (Waist:1DOF, Shoulder:1DOF, Knee:1DOf) x2

To:

3DOF x 2
3DOF x 2

Page 13

2.1 list of CPC Primitive Locator, Left fore leg, Left hind leg, Right fore leg, Right hind leg

Change:

Waist Joint
Shoulder Joint
Knee Joint

To:

J1 Joint
J2 Joint
J3 Joint

Page 16

2.2.3 Software Limitation of 4 Joints in Head

Change:

Please set the angle within the range of right roll and mouth defined in the area of tilt and pan. Pan is also symmetric on the right side. Please pay attention to the relationship between the large and small roll.

To:

Roll angle and mouth are limited to certain ranges in respective areas defined by tilt and pan angles. Pan is also symmetric on the right side. Please **note** the relationship between **the roll direction and the positive/negative sign of the roll angle**.

Page 18

2.3 Servo Gain

Change:

The following are the default servo gains in joints for ERS-210.

To:

The following are the **standard** servo gains in joints for ERS-210.

2.3 Servo Gain

The following are deleted.

```
// head tilt
// head pan
// head roll
// leg FL joint
// leg FL shoulder
// leg FL knee
// leg RL joint
// leg RL shoulder
// leg RL knee
// leg FR joint
// leg FR shoulder
// leg FR knee
// leg RR joint
// leg RR shoulder
// leg RR knee
```

Model Information ERS-220

Changes from 20020730-E-002 to 20030201-E-003

Page 17

The following are added.

2.4 Relations between the polarity of PWM and the polarity of rotation angle of joints

In OPEN-R SDK 1.1.3 r1, rotation angle of some of the joints had opposite polarity to the corresponding PWM duty. In OPEN-R SDK 1.1.3 r2, polarities of rotation angle and PWM duty are aligned for all of the joints.

Polarity of rotation angle of joint to the positive direction of PWM

	(The version of OPEN-R SDK 1.1.3)	r1	r2
PRM : /r1/c1-Joint2:j1	Neck tilt	-	+
PRM : /r1/c1/c2-Joint2:j2	Neck pan	-	+
PRM : /r1/c1/c2/c3-Joint2:j3	Neck roll	-	+
PRM : /r2/c1-Joint2:j1	Left fore Leg, J1 joint	+	+
PRM : /r2/c1/c2-Joint2:j2	Left fore Leg, J2 joint	-	+
PRM : /r2/c1/c2/c3-Joint2:j3	Left fore Leg, J3 joint	+	+
PRM : /r3/c1-Joint2:j1	Left hind leg, J1 joint	-	+
PRM : /r3/c1/c2-Joint2:j2	Left hind leg, J2 joint	-	+
PRM : /r3/c1/c2/c3-Joint2:j3	Left hind leg, J3 joint	+	+
PRM : /r4/c1-Joint2:j1	Right fore leg, J1 joint	-	+
PRM : /r4/c1/c2-Joint2:j2	Right fore leg, J2 joint	-	+
PRM : /r4/c1/c2/c3-Joint2:j3	Right fore leg, J3 joint	+	+
PRM : /r5/c1-Joint2:j1	Right hind leg, J1 joint	+	+
PRM : /r5/c1/c2-Joint2:j2	Right hind leg, J2 joint	-	+
PRM : /r5/c1/c2/c3-Joint2:j3	Right hind leg, J3 joint	+	+

Page 18

3.2 Speaker

The following are added.

ospksndMONO16K16B

Changes from 20020603-E-001 to 20020730-E-002

Page 7

1.2.2 Leg

Change:

3DOF (Waist:1DOF, Shoulder:1DOF, Knee:1DOf) x2
3DOF (Waist:1DOF, Shoulder:1DOF, Knee:1DOf) x2

To:

3DOF x 2
3DOF x 2

Page 12

2.1 list of CPC Primitive Locator, Left fore leg, Left hind leg, Right fore leg, Right hind leg

Change:

Waist Joint
Shoulder Joint
Knee Joint

To:

J1 Joint
J2 Joint
J3 Joint

Page 13

2.1 list of CPC Primitive Locator, Tail

Change:

PRM:/r6/l1-LED2:l1	Back sensor(The first from the left)
PRM:/r6/l2-LED2:l2	Back sensor(The second from the left)
PRM:/r6/l3-LED2:l3	Back sensor(The third from the left)
PRM:/r6/l4-LED2:l4	Back sensor(The third from the right)
PRM:/r6/l5-LED2:l5	Back sensor(The second from the right)
PRM:/r6/l6-LED2:l6	Back sensor(The first from the right)
PRM:/r6/l7-LED2:l7	Tail sensor(Center)
PRM:/r6/l8-LED2:l8	Tail sensor(Right)
PRM:/r6/l9-LED2:l9	Tail sensor(Left)

To:

PRM:/r6/l1-LED2:l1	Back **multi-indicator**(The first from the left)
PRM:/r6/l2-LED2:l2	Back **multi-indicator** (The second from the left)
PRM:/r6/l3-LED2:l3	Back **multi-indicator** (The third from the left)
PRM:/r6/l4-LED2:l4	Back **multi-indicator** (The third from the right)
PRM:/r6/l5-LED2:l5	Back **multi-indicator**(The second from the right)
PRM:/r6/l6-LED2:l6	Back **multi-indicator**(The first from the right)
PRM:/r6/l7-LED2:l7	Tail **light**(Center)
PRM:/r6/l8-LED2:l8	Tail **light**(Right)
PRM:/r6/l9-LED2:l9	Tail **light**(Left)

Page 15

2.2.3 Software Limitation of 4 Joints in Head

Change:

Please set the angle within the range of right roll and mouth defined in the area of tilt and pan. Pan is also symmetric on the right side. Please pay attention to the relationship between the large and small roll.

To:

A roll angle is limited to certain ranges in respective areas defined by tilt and pan angles. Pan is also symmetric on the right side. Please **note** the relationship between **the roll direction and the positive/negative sign of the roll angle**.

Page 17

2.3 Servo Gain

Change:

The following are the default servo gains in joints for ERS-220.

To:

The following are the **standard** servo gains in joints for ERS-220.

2.3 Servo Gain

The following are deleted.

// head tilt
// head pan
// head roll
// leg FL joint
// leg FL shoulder
// leg FL knee
// leg RL joint
// leg RL shoulder
// leg RL knee
// leg FR joint
// leg FR shoulder
// leg FR knee
// leg RR joint
// leg RR shoulder
// leg RR knee

Page 18

3.1 LED

Change:

PRM:/r6/l1-LED2:l1	Back sensor(The first from the left)
PRM:/r6/l2-LED2:l2	Back sensor(The second from the left)
PRM:/r6/l3-LED2:l3	Back sensor(The third from the left)
PRM:/r6/l4-LED2:l4	Back sensor(The third from the right)
PRM:/r6/l5-LED2:l5	Back sensor(The second from the right)
PRM:/r6/l6-LED2:l6	Back sensor(The first from the right)
PRM:/r6/l7-LED2:l7	Tail sensor(Center)
PRM:/r6/l8-LED2:l8	Tail sensor(Right)
PRM:/r6/l9-LED2:l9	Tail sensor(Left)

To:

PRM:/r6/l1-LED2:l1	Back **multi-indicator**(The first from the left)
PRM:/r6/l2-LED2:l2	Back **multi-indicator** (The second from the left)
PRM:/r6/l3-LED2:l3	Back **multi-indicator** (The third from the left)
PRM:/r6/l4-LED2:l4	Back **multi-indicator** (The third from the right)
PRM:/r6/l5-LED2:l5	Back **multi-indicator** (The second from the right)
PRM:/r6/l6-LED2:l6	Back **multi-indicator** (The first from the right)
PRM:/r6/l7-LED2:l7	Tail **light**(Center)
PRM:/r6/l8-LED2:l8	Tail **light**(Right)
PRM:/r6/l9-LED2:l9	Tail **light**(Left)

3.1 LED

The following are deleted.

PRM:/r1/c1/c2/c3/f1-Sensor:f1	Head sensor
PRM:/r1/c1/c2/c3/f2-Sensor:f2	Head sensor
PRM:/r1/c1/c2/c3/c4/s5-Sensor:s5	Face sensor

OPEN-R Internet Protocol Version4

Changes from 20020603-E-001

none