MINIATURE HUMANOID ROBOT

「HOAP-3」INSTRUCTION MANUAL

The First edition

								TITLE HOAP-3 INSTRUCTION MANUAL	
								DRAW. NO.	
EDIT.	DATE	DESIG.	CHECK	APPR.	DESCRIPTION				SHEET
DESIG.	041228		CHECK			APPR.			

CONTENTS

1. Outline

Since a small humanoid robot "HOAP?3" is about 60cm in height, it will be easy to treat it since it is small and lightweight, the weight of about 8kg, and if a battery is removed, and it releases the internal interface information on hardware and software, since a program is freely possible, its user is the optimal to a robot's research and development. The program which the user developed operates on RT-Linux of an instruction personal computer of operation, and communicates by the robot main part and the USB interface. Since the sensor and actuator (motor) in a robot also use the USB interface, a sensor and an actuator are easily extensible.

2. Contents

Qty	Description	Model
1	Robot body	PW09133－B001
1	Lifting jig	PW09133－D920
1	External power supply	PW09133－D910
1	CD-R	M3PW09133－B002
1	Instruction Manual	P1PW09133－A001
1	Command PC	L0PW09133－0001

3. System Configuration

3. 1 Overall System drawing

The whole system composition is shown in Fig. 3.1-1 and Fig. 3.1-2. Selection of cable Mohd and radio Mohd of this system is attained. In cable Mohd, communication of the high speed and real time which used USB is attained between Exterior PC and a robot main part. An external power supply (24V) is used for the power supply of a robot main part. OS of Exterior PC is RT-Linux (*), can use USB among robots and can perform feedback control on real time. Let the battery of the robot inside of the body be a power supply in radio Mohd. And real-time control is performed in the inside PC carried in a robot's back. From Exterior PC, control of this inside PC is wireless LAN, and is remote accessed.

(*) The user needs to install RT-Linux. Refer to Appendix 3.1 for the hardware requirement and the installation method of RT-Linux.

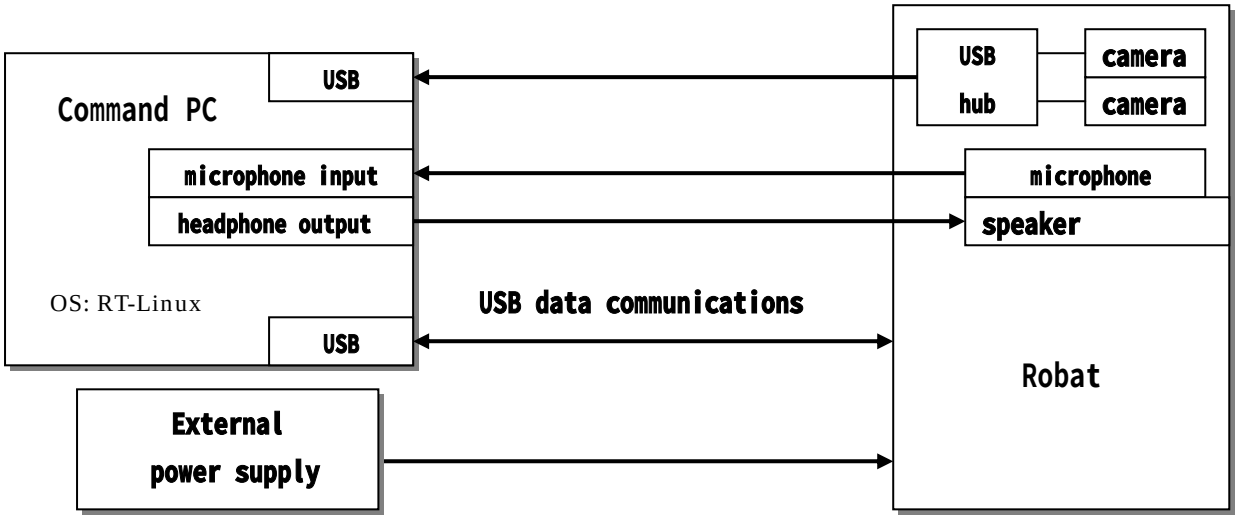


Fig.3.1-1 The system configuration in cable mode

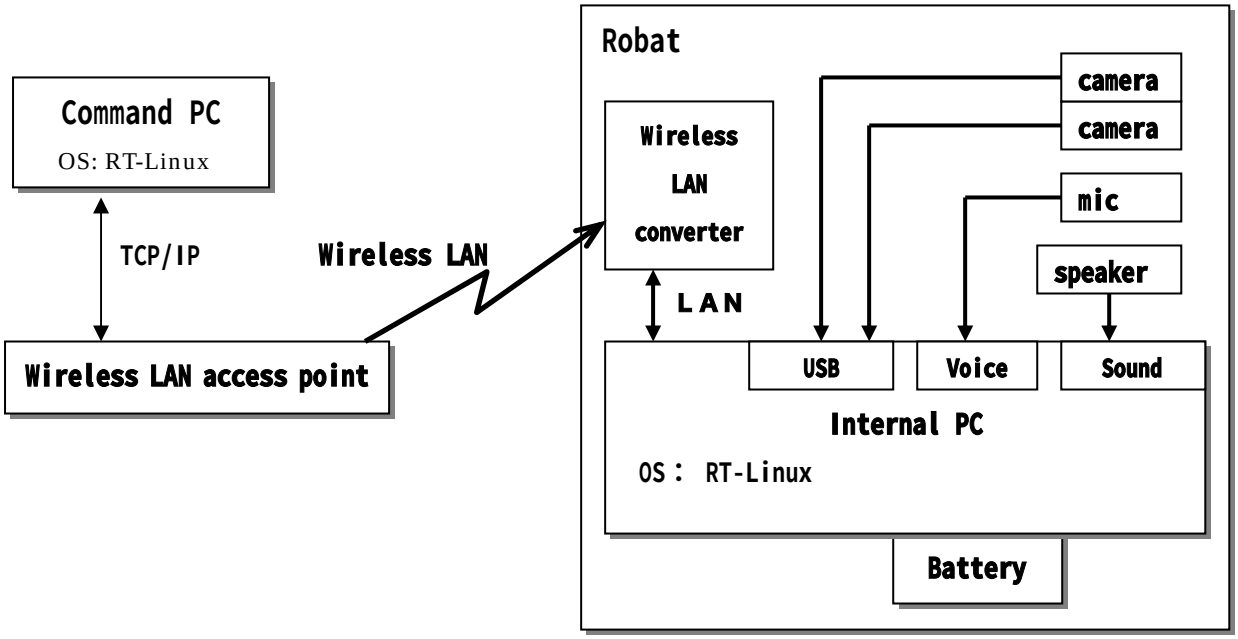


Fig3.1-2 The system configuration in radio mode

3．2　Connection method

This chapter explains the connection method of system configuration apparatus.

(1) Cable mode

①Please confirm that both two power switches (a motor power switch and logic power switch) in the robot main part upper part are turned "off."

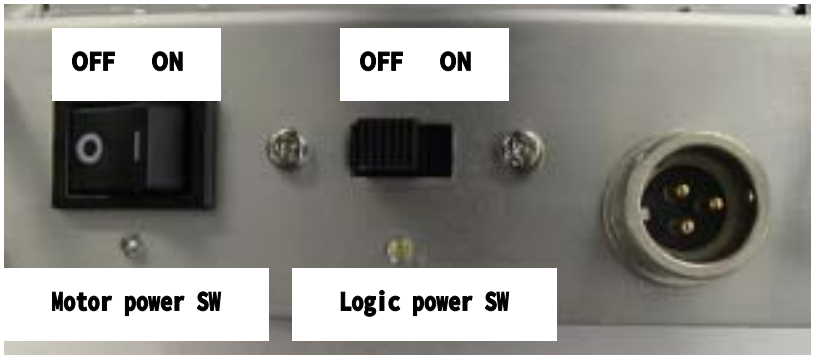


Fig3.2-1　Two power switches

② **I connect between Command PC and robot main parts with a USB telecommunication cable. Please connect Robot side to the robot USB port.　Please refer to Fig S1.2-3 of Appendix 1-3 about the robot USB port.**
Please connect Command PC side to the USB port on the back of a personal computer.

③ **I connect between external power supplies with a robot with an exclusive power cable. Please connect the connection with a robot to an external power supply input port. Please refer to the Fig.S1.2-4 of Appendix 1-3 about an external power supply input port. Please connect an external power supply side to the connector in the front side of a power supply.**

④ **Finally I connect an external power supply to AC wall socket, and all connection is completed (the connection about other instruction personal computers of operation should follow the instructions manual of a personal computer).**

(2) Wireless mode

① An external power supply is OFF and a back cover is removed after checking that the motor power switch and the logic power switch are turned off. Refer to Fig.S1.2 of Appendix 1-2 for how to remove a back cover.

② The USB cable from Inside PC is connected to a robot USB port.

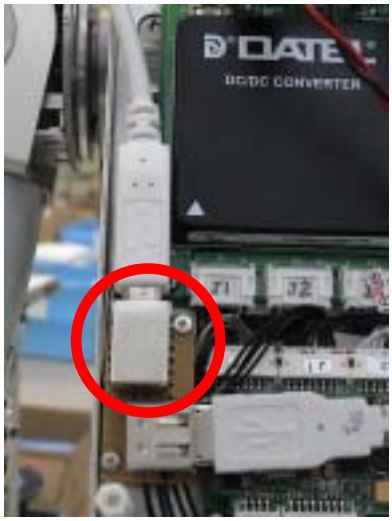


Fig.3.2-2　robot USB port

③ The power supply line of a wireless LAN converter is connected. (Refer to Fig. 3.2-3.)

Fig.3.2-3 The power supply line of a wireless LAN converter

④ The power supply of Inside PC is connected. (Refer to Fig. 3.2-4)

Fig.3.2-4 Power supply of Inside PC

⑤ A battery is connected. Refer to the "Fig.1.3 battery exchange method" of Appendix 1-4 .

Reference :

The following methods are effective in order to prevent consumption of a battery. Till just before operation, an external power supply is connected without connecting a battery. And the full charged battery is connected and the connector of an external power supply is removed after that just before operation. If an external power supply connector is removed, please turn off the power switch by the side of an external power supply. Moreover, after completing operation, an external power supply connector is connected and the power switch of an external power supply is turned on. And a battery is removed. (Even if the external power supply and the battery are connected simultaneously, since a robot is only supplied from a side with high voltage, it is satisfactory.)

⑥ Since connection is an end above, please follow the "Fig.S3.2 radio operation procedure" of Appendix 3-11.

- Please remove the power supply line of a wireless LAN converter and Inside PC, and also remove a battery, after turning OFF all a robot's power supplies, when returning to cable mode from radio mode. Moreover, please return a USB cable to the state in cable mode similarly. If the power supply line of Inside PC is connected, Inside PC will rise at the time of subsequent power supply ON, if a power supply is turned off in this state, a power supply will be suddenly shut off by Inside PC and the contents of the CF card may be destroyed.

3.3 Major parts position

The position of main parts is shown in Fig.3.3-1 - 3.3-3.

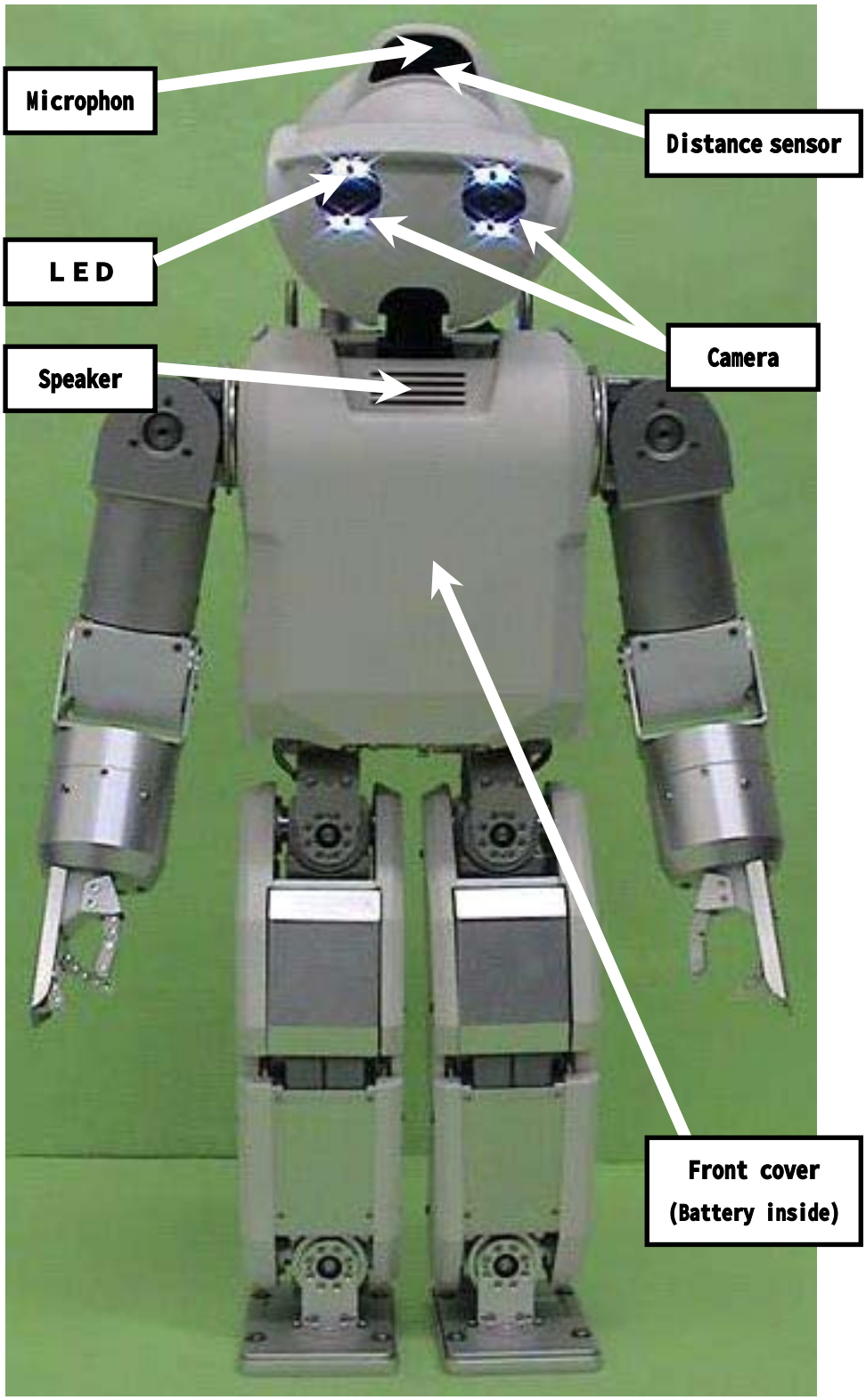


Fig.3.3-1 The front of robot

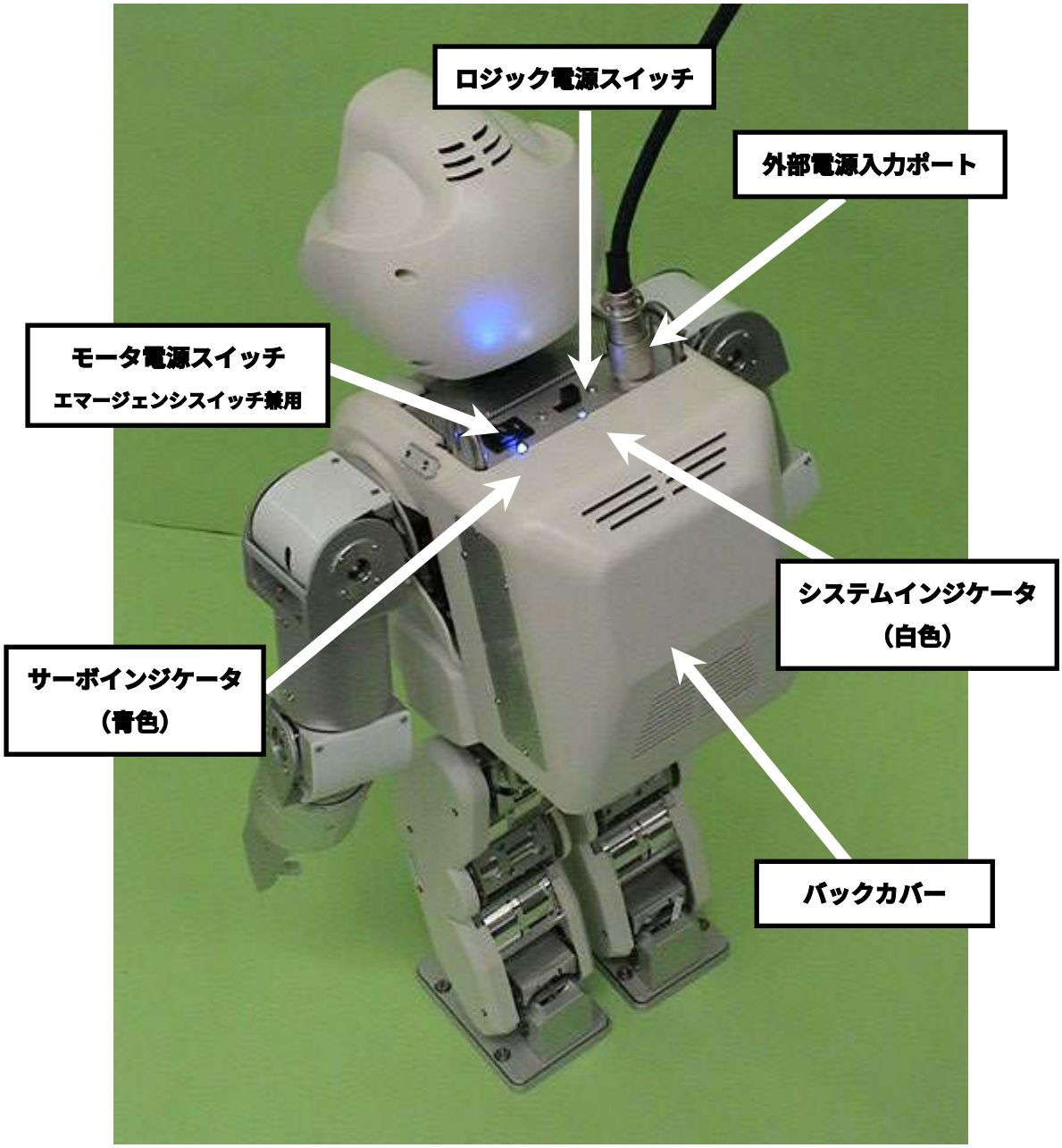


Fig.3.3-2 Robot back

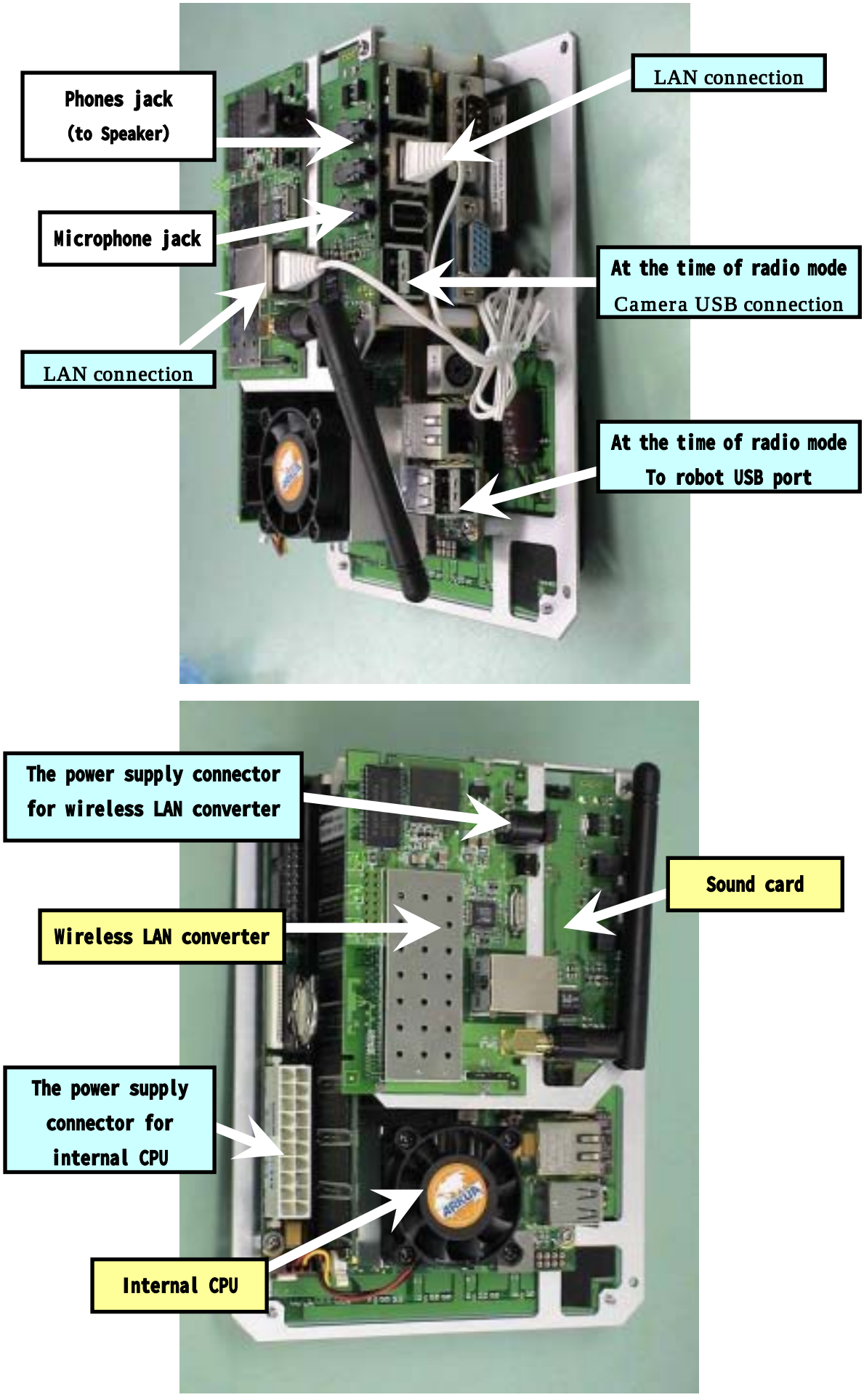


Fig.3.3-3 Inside of robot

4 Cautionary Items

Please read this "Cautionary Items" before use this robot system.

Please be sure to keep it at the place seen always.

4．1 Precautions and handling

HOAP-3 is a highly advanced programming unit with many degrees of freedom joints. Please follow all instructions and cautionary items to avoid breakage,other troubles or possible operator injury.

Although classified into "danger", "warning", and "cautions",please be sure to follow notes shown here, since the important contents with which all are related safely are indicated.

Danger : That it is assumed to be that a risk of a user getting death or seriously injured draws near and arises when handling is mistaken.

Warning : That a possibility that a user will get death or seriously injured is assumed to be when handling is mistaken.

Cautions: The thing it is assumed to be that a user gets a serious injury or slightly injured when handling is mistaken, and the thing which material damage generates. Or when handling is mistaken and it is assumed that the quality and the reliability of a product are spoiled.

4.1.1 Power supply and battery

Danger

- Do not use non-standard voltage.
- Do not use non-specified battery charger.
  It becomes a liquid leak and the cause which makes it generate heat and explode about a battery.
- Do not leave unattended during charging.
- A battery is not thrown in in fire and it does not heat.
- The (+) and (-) terminal of a batteryare not connected with metal, such as wire. Moreover, it does not carry together with a metal necklace, a hairpin, etc.
- A battery is not disassembled and converted. Generation of heat, ignition, and the liquid of strong alkalinity emit and are dangerous.
- Don't solder to a battery directly.
- As for the battery, it opts for direction of plus and minus.
  It does not connect by force. Please confirm direction of plus and minus.
- This battery is only use for HOAP robot. Don't use this battery to another use.
- When the liquid of a battery goes into an eye, wash enough with beautiful water immediately and undergo a doctor's medical treatment.

Warning

- A battery is not attached to water, sea water, etc.A terminal portion is not wet. It becomes the cause of heat and the rust of a terminal etc.
- Charge is stopped, when charge is not completed, even if it exceeds predetermined charge.
- Since injury may be caused on the skin when the liquid of a battery adheres to the skin or clothes, please wash away with beautiful water immediately.
- A battery coating tube is not removed or a crack is not given. It becomes a liquid leak and the cause which makes it generate heat and explode about a battery.
- Don't use the battery, when a battery carries out a liquid leak or it has noticed differing from discoloration.
- Don't use the battery and leave in hot places, such as a strong place of direct rays, in the car, and the front of a stove etc.

Caution

- Don't give a shock strong against a battery or don't throw it.
- Be sure to perform use (electric discharge) of a battery in 0~50℃.
- Charge the battery in 0～40℃.
- Keep a battery in a place with little humidity of 0～30℃.
- Don't charge in having got the battery cold and cold outdoors (0℃ or less) .
- When robot is not in operation, remove the battery and make sure that all power switches are in the off position to avoid battery discharge.
- When not in use, make sure that power switch is in the OFF position
- Don't use the battery in unusual.
- Motion sufficient by battery drive can't be performed depending on operation.
- When battery residual quantity decreases, if large current flows, CPU may reboot. Be fully careful.

<Abandonment of a battery>

Used nickel hydoride batteries are precious resources. Please bring at a nickel hydoride battery recycling cooperation store.

4.1.2 Main Body

Danger

- Do not touch and keep a way inside of robot operation range.
  (for emergency stop, switch off motor power in front side)
- Keep clear when servo motors are turned ON.
- Be careful hands not to be caught between activated portion.
- Don't use a robot in an area known to have an electro-magnetic interference, static electricity discharge or possible radio frequency interference.
- The robot is likely to experience faulty operation and can be dangerous to operate under such conditions.
- Set up the joint initial setting in the right position, If it does not do so, the robot may do unusual operation and this will cause damage to the robot.

Warning

- Robot is designed to be operated in a clean, or laboratory environment.

  Do not attempt to use where dampness, dust, or unfavorable temperature conditions are present.

- The motor can become hot during operation.

  Make sure that motor power is off and motors are cooled properly prior to touching

- Do not use robot in a fashion where excessive torque is applied to joints, or beyond the allowable joint torque. This includes manually moving joints. This may cause damage to the robot.

Caution

- Tension on timing belts located in the legs should be maintained as described in the appendix. Excessive overweight can cause belts to lose tension.
- If it is left in the state where a motor does not arrive at a target position, a motor will become high temperature and this will cause damage to the robot.

4.1.3 Programming

Danger

- Knowledge of RT-Linux and robot movement principles will make programming of the robot easier
- Make sure that programmed moves are made relative to the soft-limit when developing your program
- The optimum joint motion is accomplished by using minute steps. When using small steps, the robot makes smooth movements. It is very dangerous when the movement command value between each step is increased too much. When a movement command is issued, the servos will move from current position to the newly commanded position at high speed. If the new position is greatly different from the current position it can result in a dangerous situation. It is recommended, and is the responsibility of the user to develop safeguards in their software that, within the boundary of their experiments, will reject commands that instruct movements which greatly vary from the current range and position.
- It is very dangerous to use this robot with speed control mode.

  Consult our company when you use speed control mode.

4. 2 Warranty

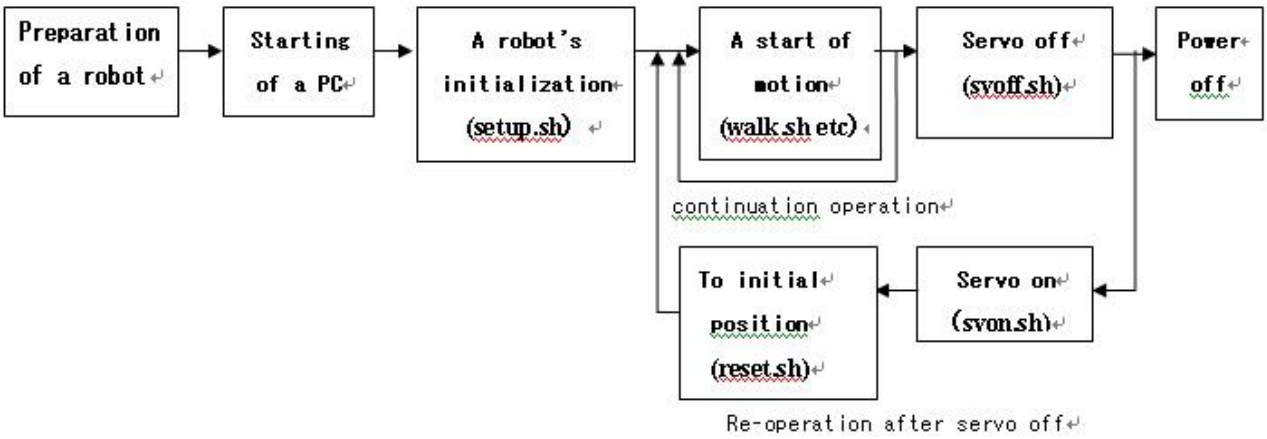
It is chargeable except for the initial defect.

This robot's appearance and specification may be changed without a preliminary announcement.

5. Confirmation of Operation

5.1 Operation Flow

The steps to operate the robot are shown in the following process flow chart. These steps are necessary in order to initialize the robot parameters before an operation begins.



5.2 Example of How to Run a Sample Motion Program

The following explains a sample operation procedure for the wired mode. Please get used to a robot's operation procedure through this chapter.

5.2.1 Preparation of the ROBOT

(1) Please check that the logic power switch and the motor power switch are turned off. When not turned off, please turn OFF at the order of a motor power switch -> logic power switch.

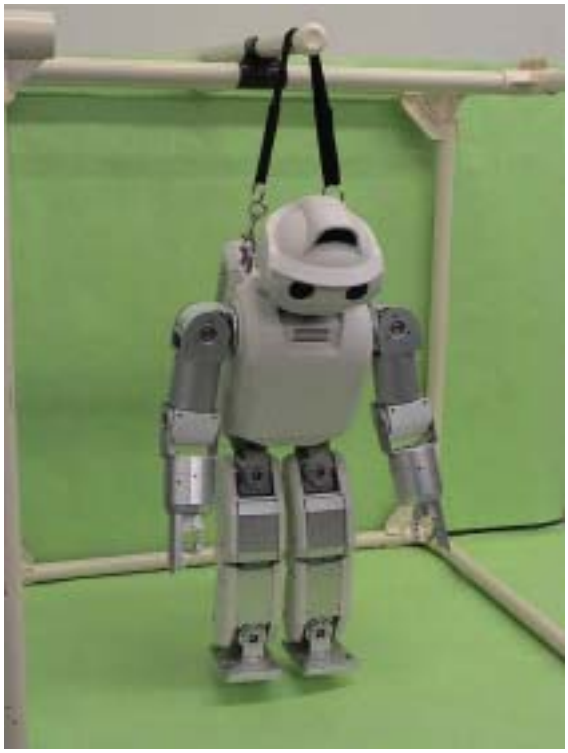
(2) Please check that the external power supply is turned off. When not turned off, please turn OFF.

(3) A robot is hung to a Hang stand implement. (Refer to Fig. 5.2-1)

(4) Please connect an exclusive USB cable to a robot USB port and the USB port on the personal computer back side.

(5) About the cable for exclusive 24V power supplies, it is a robot's external power supply. Please connect with an input port and an external power supply.

Cautions: Please do cable connection work after checking that 24V power supply is turned off.



5.2.2 Starting of a PC

(6) Turn On the PC.

(7) After Red Hat Linux starts, you will see the following prompt: and input following.

a. local host login : **root (Return)**

b. password : **default (Return)**

(8) Start terminal emulator for GNOME.

Input: **#startx (Return)**

(9) Start to『terminal emulator for GNOME』

(10) Change directory.

Input **#cd /usr/local/hoap3/modules (Return)**

(11) RT-Linux is started.

Input **#rtlinux start rt_ctlmodule.o（Return）**

When RT-Linux has started normally, the following will be displayed (+).

（＋）rt_ctlmodule

:

If it is displayed (－) , Reboot the PC and restart from (6) .

(12) It is input as follows, and a communication module with the robot is started

 # ../bin/rt_ctlapp (Return)

(13) Turn on the logic and motor power switches of the robot.

(14) Reset the USB devices. Input the "**r**" command.

0,00,00$ **r (Return)**

(15) If it is continuing pushing a Return key , the command PC starts the recognition of the USB device.

Repeat it until it is indicated below.

5, 27, xx $ (xx isn't fixed)

5.2.3 A robot's initialization

(16) Start another terminal emulator for GNOME.

This emulator window is named The window 2 , and the window currently opened from before is named a window 1.

(17) In the window2, input as follows, and change directory to /usr/local/hoap/data .

#cd /usr/local/hoap3/data (Return)

(18) Where a robot is hung, please check that all joints are near the initial specified position. If the position is shifted greatly, initial-setting operation will become impossible normally and will become an error on the way.

5.2.4 Starting point return program

(19) An automatic zero return program is started. Please input as follows in a window 2.

 # sh setup.sh (Return)

Note! After this command transmission, automatically, a servo is set to ON and performs a starting point search. Since a motor axis operates, be careful not to pinch a hand etc.

5.2.5 Sample walk operation start

(20) Sample walk operation is started. Please input the following in a window 2.

 # sh walk.sh (Return)

※By transmitting this command, a robot begins sample walk operation.

Operation performed to the next after a robot's end of operation,

a) When the same operation is performed again -> Please carry out from (20).

b) When ending, or when servo off is carried out once -> please move on to "5.2.6 servo off.

5.2.6 Servo off

(21) Servo off. Since servo control of all the robot's joint angles is turned off, please hang a robot and be sure to hang him to a Hang stand. If servo Off [without hanging], a robot falls and there is a risk of damaging. Please input the following in a window 2.

 # sh svoff.sh (Return)

In order to operate a robot again, without turning off the power, please input as follows next.

sh svon.sh **(Return)**
sh reset.sh **(Return)**
sh walk.sh **(Return)**

5.2.7 Power off

(22) A robot's motor power supply -> please turn OFF a power supply at the order of a logic power supply.

(23) Please turn OFF an external power supply.

(24) Please close a window 2.

(25) Please input and quit "q" as follows in a window 1.

 # rtlinux stop rt_ctlmodule.o (Return)

(26) Red Hat software Linux is stopped.

 # shutdown –h now (Return)

(27) Please turn OFF the power supply of a personal computer.

Above, experiment operation of a robot is an end.

[Wireless mode]

Except for the point of remote accessing from an instruction personal computer of operation, it becomes the same operation as the above using wireless LAN. As for "3.2 The connection method" and the procedure method of operation, refer to " S3.2 radio operation procedure" for the connection method.

6. Mechanical

6.1 Joint composition and the definition of the angle

This robot has a total of 28 joints in two arms of 6 flexibility, two legs of 6 flexibility, and the head of 3 flexibility and the body part of 1 flexibility. It is rotation flexibility altogether and a leg, an arm, and the waist are serial links. The name of a joint and flexibility distribution have become as it is shown in Fig. 6.1-1 and Table 6.1-1. in addition, [] of the joint name of front Naka -- an inside numerical value -- the order from 1 -- the machine from a root -- structural connection ranking is shown. Moreover, the joint angle which determines a robot's posture is defined by Fig. 6.1-2, 6.1.-3, the link parameter of Table 6.1-2, the joint coordinate system, and DH parameter, and **all joint instructions to a robot should follow this definition.**

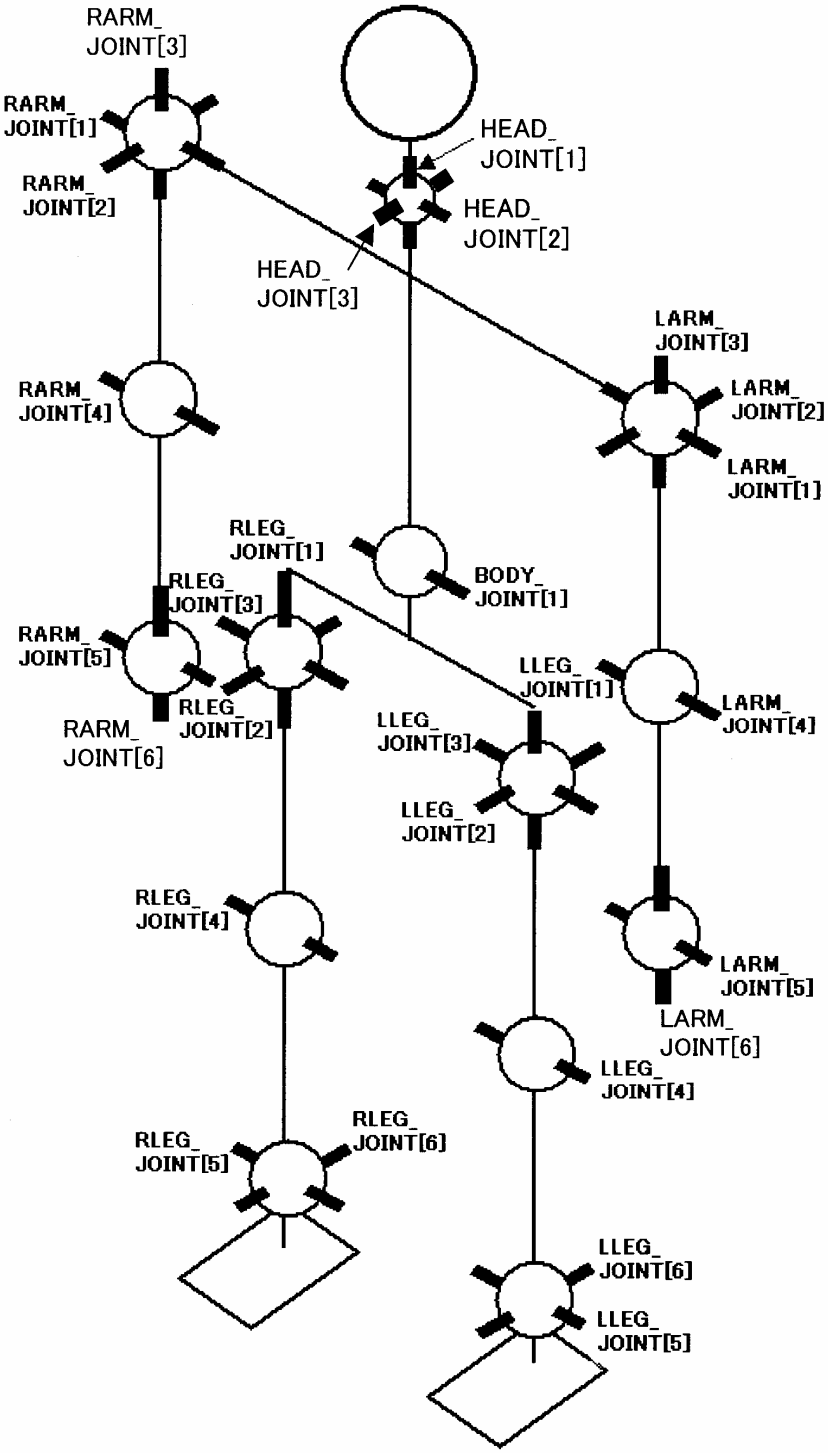


Fig.6.1-1 Joint name

Table6.1-1 Joint name

Joint name	Flexibility	Device ID
HEAD_JOINT[1]	Head torsion	22
HEAD_JOINT[2]	Head pitch	22
HEAD_JOINT[3]	Head roll	22
BODY_JOINT[1]	Waist pitch	21
RLEG_JOINT[1]	Right hip joint torsion	1
RLEG_JOINT[2]	Right hip joint roll	2
RLEG_JOINT[3]	Right hip joint pitch	3
RLEG_JOINT[4]	Right knee	4
RLEG_JOINT[5]	Right Ankle pitch	5
RLEG_JOINT[6]	Right Ankle roll	6
RARM_JOINT[1]	Right shoulder Pitch	7
RARM_JOINT[2]	Right shoulder roll	8
RARM_JOINT[3]	Right shoulder torsion	9
RARM_JOINT[4]	Right elbow	10
RARM_JOINT[5]	Right fingers open/close	23
RARM_JOINT[6]	Right hand torsion	23
LLEG_JOINT[1]	Left hip joint torsion	11
LLEG_JOINT[2]	Left hip joint roll	12
LLEG_JOINT[3]	Left hip joint pitch	13
LLEG_JOINT[4]	Left knee	14
LLEG_JOINT[5]	Left Ankle pitch	15
LLEG_JOINT[6]	Left Ankle roll	16
LARM_JOINT[1]	Left shoulder Pitch	17
LARM_JOINT[2]	Left shoulder roll	18
LARM_JOINT[3]	Left shoulder torsion	19
LARM_JOINT[4]	Left elbow	20
LARM_JOINT[5]	Left fingers open/close	23
LARM_JOINT[6]	Left hand torsion	23

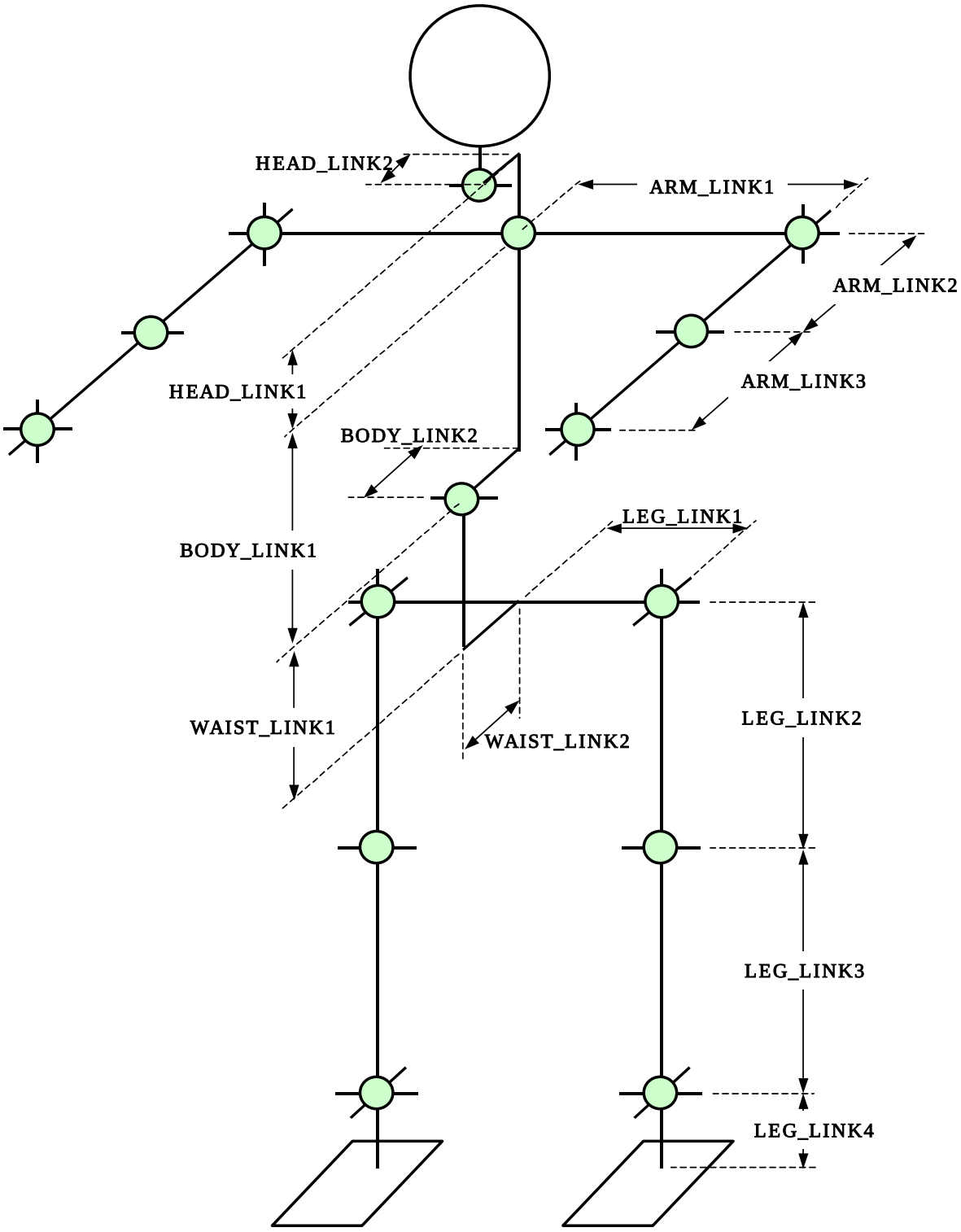


Fig.6.1-2 The parameter definition of the link length

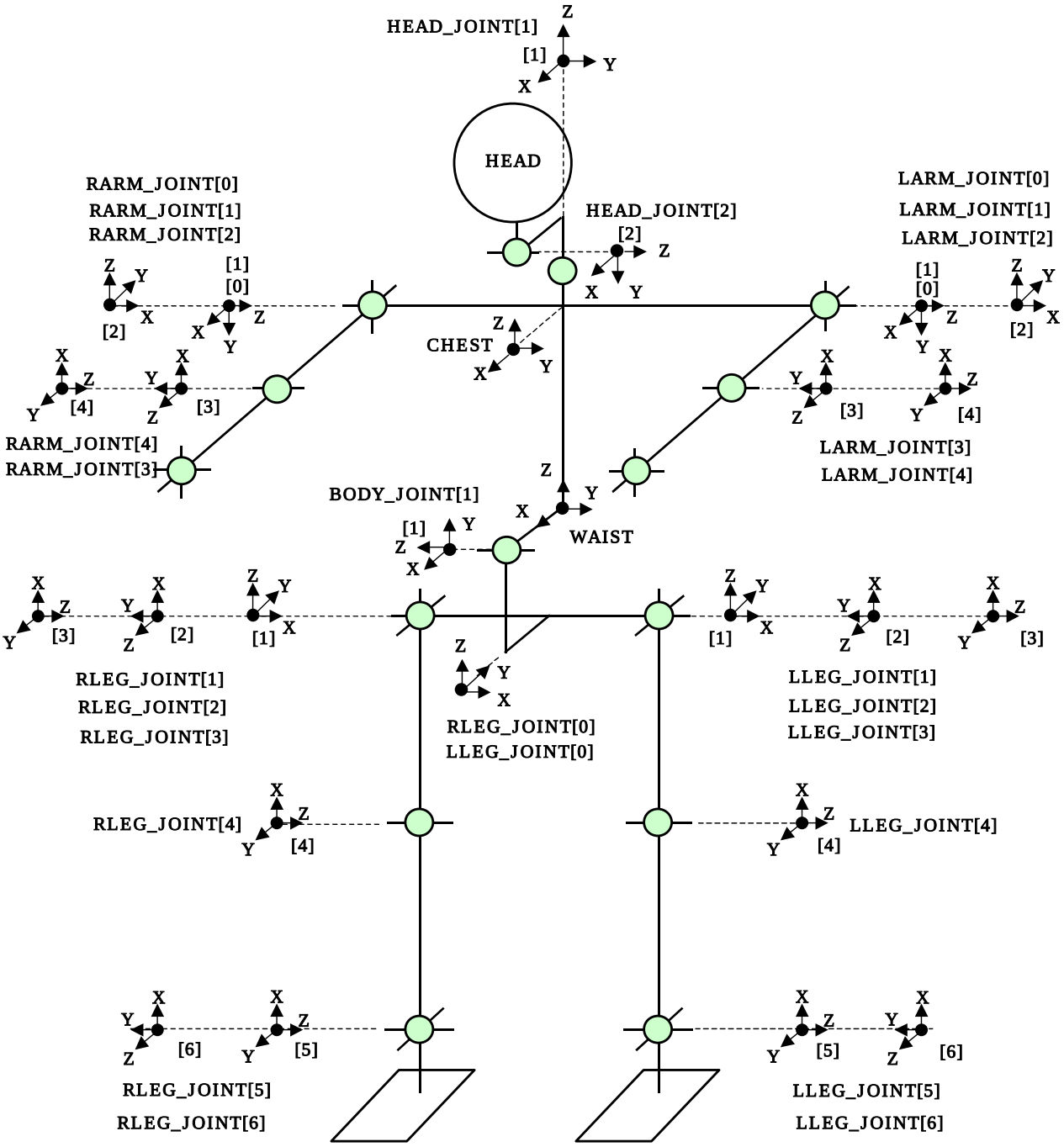


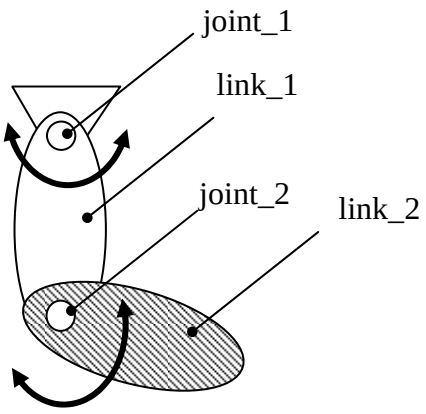
Fig.6.1-3　DH parameter definition coordinate（θ[i]=0 Posture）

Table.6.1-2 DH Parameter (theta[i] =0 is the posture of Fig.6.1-3

	a[I-1]	α[I-1]	d [i]	θ [i]
	(m)	(deg)	(m)	(deg)
CHEST	-BODY_LINK2	0	BODY_LINK1	0
HEAD_JOINT[1]	0	0	HEAD_LINK1	0
HEAD_JOINT[2]	HEAD_LINK2	-90	0	0
ARM_JOINT[0]	0	-90	(*)ARM_LINK1	0
ARM_JOINT[1]	0	0	0	θ [1]
ARM_JOINT[2]	0	90	0	θ [2] +90
ARM_JOINT[3]	0	90	ARM_LINK2	θ [3] +90
ARM_JOINT[4]	0	90	0	θ [4]
BODY_JOINT[1]	0	90	0	0
LEG_JOINT[0]	-WAIST_LINK2	-90	-WAIST_LINK1	90
LEG_JOINT[1]	(*)LEG_LINK1	0	0	θ [1]
LEG_JOINT[2]	0	90	0	θ [2] +90
LEG_JOINT[3]	0	90	0	θ [3]
LEG_JOINT[4]	-LEG_LINK2	0	0	θ [4]
LEG_JOINT[5]	-LEG_LINK3	0	0	θ [5]
LEG_JOINT[6]	0	-90	0	θ [6]
	(*)Please give as a negative value at the time of a right half the body.			
	WAIST	－＞	CHEST	
ARM_LINK1	0.111	m		
ARM_LINK2	0.111	m		
ARM_LINK3	0.171	m		
LEG_LINK1	0.039	m		
LEG_LINK2	0.105	m		
LEG_LINK3	0.105	m		
LEG_LINK4	0.040	m		
BODY_LINK1	0.125	m		
BODY_LINK2	0.035	m		
HEAD_LINK1	0.103	m		
HEAD_LINK2	0.015	m		
WAIST_LINK1	0.055	m		
WAIST_LINK2	0.035	m		

6.2 Mass Property

mass property of the necessary link is in the attached sheet to simulate the dynamics of the robot. Mass property is defined in the link unit that it belongs to the degree of each joint freedom in accordance with the joint coordinate of the DH parameter .A link for example to belong to joint_1 in case as a ground plan becomes link_1, and mass property of link_1 by the coordinate of joint_1 is defined as it.



And, three kinds of mass property show it about the BODY link which is the body trunk of the robot by the difference in the loading thing of the cable mode and the wireless mode. Select and simulate mass property fitted to the configuration of an actual opportunity

6.3 Conversion of a joint angle and speed

Control of a robot's joint angle or speed uses as an instruction value what changed into the increase and decrease (speed) of a value of the pulse count value (position) of the encoder of a joint, or the pulse count value around time the position defined by DH parameter and speed. It is answered to all also of the position of the present joint, and the information on speed by the robot with this value. The incremental encoder is used for an encoder. Therefore, PURISETTO [the counter used as a position instruction value / a joint angle / a known posture] so that it can change into the joint angle absolute value defined by DH parameter (or reverse conversion). PURISETTO [specifically / robot / an automatic starting point return program / a joint angle] automatically. If PURISETTO completion is carried out, a joint angle and speed are convertible for the definition semi- じた value of DH parameter by the conversion formula of Table 6.3-1 and 6.3-2 (or reverse conversion).

Table 6.3-1 Conversion of DH parameter definition joint angle and position instruction value

θd	DH definition joint angle(deg)
P	Position instruction value(pulse)
A	Conversion factor(pulse/deg)
Note)	P is 2 bytes of integer with a mark
Conversion type:	**P=A× θd**
Joint name	A(pulse/deg)
RLEG__JOINT[1]	209
RLEG__JOINT[2]	209
RLEG__JOINT[3]	-209
RLEG__JOINT[4]	209
RLEG__JOINT[5]	209
RLEG__JOINT[6]	-209
RARM__JOINT[1]	209
RARM__JOINT[2]	209
RARM__JOINT[3]	-209
RARM__JOINT[4]	-209
LLEG__JOINT[1]	209
LLEG__JOINT[2]	209
LLEG__JOINT[3]	209
LLEG__JOINT[4]	-209
LLEG__JOINT[5]	-209
LLEG__JOINT[6]	-209
LARM__JOINT[1]	-209
LARM__JOINT[2]	209
LARM__JOINT[3]	-209
LARM__JOINT[4]	209
BODY__JOINT[1]	209

Table 6.3-2 Conversion of DH parameter definition joint speed and speed instruction value

ωd	DH definition joint speed (deg/s)
V	Speed instruction value(pulse/ms)
B	Conversion factor(pulse/ms/(deg/s))
note)	V is 2 bytes of integer with a mark
Conversion type:	**V=B×ωd**
Joint name	B(pulse/ms/(deg/s))
RLEG__JOINT[1]	0.209
RLEG__JOINT[2]	0.209
RLEG__JOINT[3]	-0.209
RLEG__JOINT[4]	0.209
RLEG__JOINT[5]	0.209
RLEG__JOINT[6]	-0.209
RARM__JOINT[1]	0.209
RARM__JOINT[2]	0.209
RARM__JOINT[3]	-0.209
RARM__JOINT[4]	-0.209
LLEG__JOINT[1]	0.209
LLEG__JOINT[2]	0.209
LLEG__JOINT[3]	0.209
LLEG__JOINT[4]	-0.209
LLEG__JOINT[5]	-0.209
LLEG__JOINT[6]	-0.209
LARM__JOINT[1]	-0.209
LARM__JOINT[2]	0.209
LARM__JOINT[3]	-0.209
LARM__JOINT[4]	0.209
BODY__JOINT[1]	0.209

6.4 Joint allowable movement range

The joint movement range (DH parameter angle minimum angle - a maximum angle) that it is permitted after pre- sets joint counter of the robot becomes a table 6.4-1. Software limit of movement range can be set up in the robot. Be sure to set up software limit in less than this movement range by counter value for the prevention of damage and the safety. And, software limit is set up by the maximum and minimum of counter value. Over shoot of the control to the range of a joint angle to use in the actual practical use, and take the inside of 1deg of the maximum-minimum value of the table 6.4-1.

Sample: The utility movement ranges of RLEG_JOINT [1] become -90 - 30deg.

Table 6.4-1 Joint allowable movement range

(counter value is value with a thing after the pre-set completion.)

Joint name	Flexibility	DH parameter minimum		DH parameter maximum		Motor type	Device ID
		DH angle (deg)	Counter value (Decimal)	DH angle (deg)	Counter value (Decimal)		
RLEG_JOINT[1]	Right hip torsion	-91	-19019	31	6479	TYPE-2	1
RLEG_JOINT[2]	Right hip roll	-31	-6479	21	4389	TYPE-3	2
RLEG_JOINT[3]	Right hip pitch	-82	17138	71	-14839	TYPE-2	3
RLEG_JOINT[4]	Right knee	-1	-209	130	27170	TYPE-3	4
RLEG_JOINT[5]	Right ankle pitch	-61	-12749	61	12749	TYPE-3	5
RLEG_JOINT[6]	Right ankle roll	-25	5225	25	-5225	TYPE-2	6
RARM_JOINT[1]	Right shoulder pitch	-91	-19019	151	31559	TYPE-2	7
RARM_JOINT[2]	Right shoulder roll	-96	-20064	1	209	TYPE-2	8
RARM_JOINT[3]	Right shoulder	-91	19019	91	-19019	TYPE-2	9
RARM_JOINT[4]	Right elbow	-115	24035	1	-209	TYPE-2	10
LLEG_JOINT[1]	Left hip torsion	-31	-6479	91	19019	TYPE-2	11
LLEG_JOINT[2]	Left hip roll	-21	-4389	31	6479	TYPE-3	12
LLEG_JOINT[3]	Left hip pitch	-82	-17138	71	14839	TYPE-2	13
LLEG_JOINT[4]	Left knee	-1	209	130	-27170	TYPE-3	14
LLEG_JOINT[5]	Left ankle pitch	-61	12749	61	-12749	TYPE-3	15
LLEG_JOINT[6]	Left ankle roll	-25	5225	25	-5225	TYPE-2	16
LARM_JOINT[1]	Left shoulder pitch	-91	19019	151	-31559	TYPE-2	17
LARM_JOINT[2]	Left shoulder roll	-1	-209	96	20064	TYPE-2	18
LARM_JOINT[3]	Left shoulder torsion	-91	19019	91	-19019	TYPE-2	19
LARM_JOINT[4]	Left elbow	-115	-24035	1	209	TYPE-2	20
BODY_JOINT[1]	Waist pitch	-1	209	90	-18810	TYPE-3	21
HEAD_JOINT[1]	Head torsion	-60	-	60	-	RC-サーボ	22
HEAD_JOINT[2]	Head pitch	-45	-	15	-	RC-サーボ	22
HEAD_JOINT[3]	Head roll	-15	-	15	-	RC-サーボ	22
RARM_JOINT[5]	Right fingers	-60	-	60	-	RC-サーボ	23
RARM_JOINT[6]	Right hand torsion	-90	-	90	-	RC-サーボ	23
LARM_JOINT[5]	Left fingers	-60	-	60	-	RC-サーボ	23
LARM_JOINT[6]	Left hand torsion	-90	-	90	-	RC-サーボ	23

※ The movement range of these joints may decrease, because of overlap.

6.5 Joint allowable torque

The allowable torque of the joint of the robot shows a table 6.5-1. Be careful not to exceed this value when the back drive operation of the joint by manual in direct acting servo off or the time of the movement of the robot.

Table 6.5-1 Joint Allowable torque

item	joint Name	Flexibility	Joint Permission torque [N·m]		Motor type
			Average	Peak	
1	RLEG_JOINT[1]	Right hip torsion	1.5	3	TYPE-2
2	RLEG_JOINT[2]	Right hip roll	2	4.5	TYPE-3
3	RLEG_JOINT[3]	Right hip pitch	1.5	3	TYPE-2
4	RLEG_JOINT[4]	Right knee	2	4.5	TYPE-3
5	RLEG_JOINT[5]	Right ankle pitch	2	4.5	TYPE-3
6	RLEG_JOINT[6]	Right ankle roll	1.5	3	TYPE-2
7	RARM_JOINT[1]	Right shoulder pitch	1.5	3	TYPE-2
8	RARM_JOINT[2]	Right shoulder roll	1.5	3	TYPE-2
9	RARM_JOINT[3]	Right shoulder	1.5	3	TYPE-2
10	RARM_JOINT[4]	Right elbow	1.5	3	TYPE-2
11	LLEG_JOINT[1]	Left hip torsion	1.5	3	TYPE-2
12	LLEG_JOINT[2]	Left hip roll	2	4.5	TYPE-3
13	LLEG_JOINT[3]	Left hip pitch	1.5	3	TYPE-2
14	LLEG_JOINT[4]	Left knee	2	4.5	TYPE-3
15	LLEG_JOINT[5]	Left ankle pitch	2	4.5	TYPE-3
16	LLEG_JOINT[6]	Left ankle roll	1.5	3	TYPE-2
17	LARM_JOINT[1]	Left shoulder pitch	1.5	3	TYPE-2
18	LARM_JOINT[2]	Left shoulder roll	1.5	3	TYPE-2
19	LARM_JOINT[3]	Left shoulder torsion	1.5	3	TYPE-2
20	LARM_JOINT[4]	Left elbow	1.5	3	TYPE-2
21	BODY_JOINT[1]	Waist pitch	2	4.5	TYPE-3
22	HEAD_JOINT[1]	Head torsion	0.87	1.12	RCサーボ
23	HEAD_JOINT[2]	Head pitch	0.87	1.12	RCサーボ
24	HEAD_JOINT[3]	Head roll	0.87	1.12	RCサーボ
25	RARM_JOINT[5]	Right fingers	0.28	0.36	RCサーボ
26	RARM_JOINT[6]	Right hand torsion	0.28	0.36	RCサーボ
27	LARM_JOINT[5]	Left fingers	0.28	0.36	RCサーボ
28	LARM_JOINT[6]	Left hand torsion	0.28	0.36	RCサーボ

6.6 Sensors

6.6.1 A position of sensor loading

The sensors carried in the robot are

・Posture sensor 1pc (a gyroscope sensor and acceleration sensor)

・The foot bottom sensor 2 pcs

(it is one piece at a time to four corners of a pair-of-shoes bottom.)

・Grip sensor 2 pcs (It carries at a time in the one thumb of a hand on either side)

・USB camera 2pcs (An eye on either side)

A posture sensor combines an acceleration sensor and a gyroscope sensor. These loading positions are shown in Fig.6.6-1.

Acceleration sensor appearance position

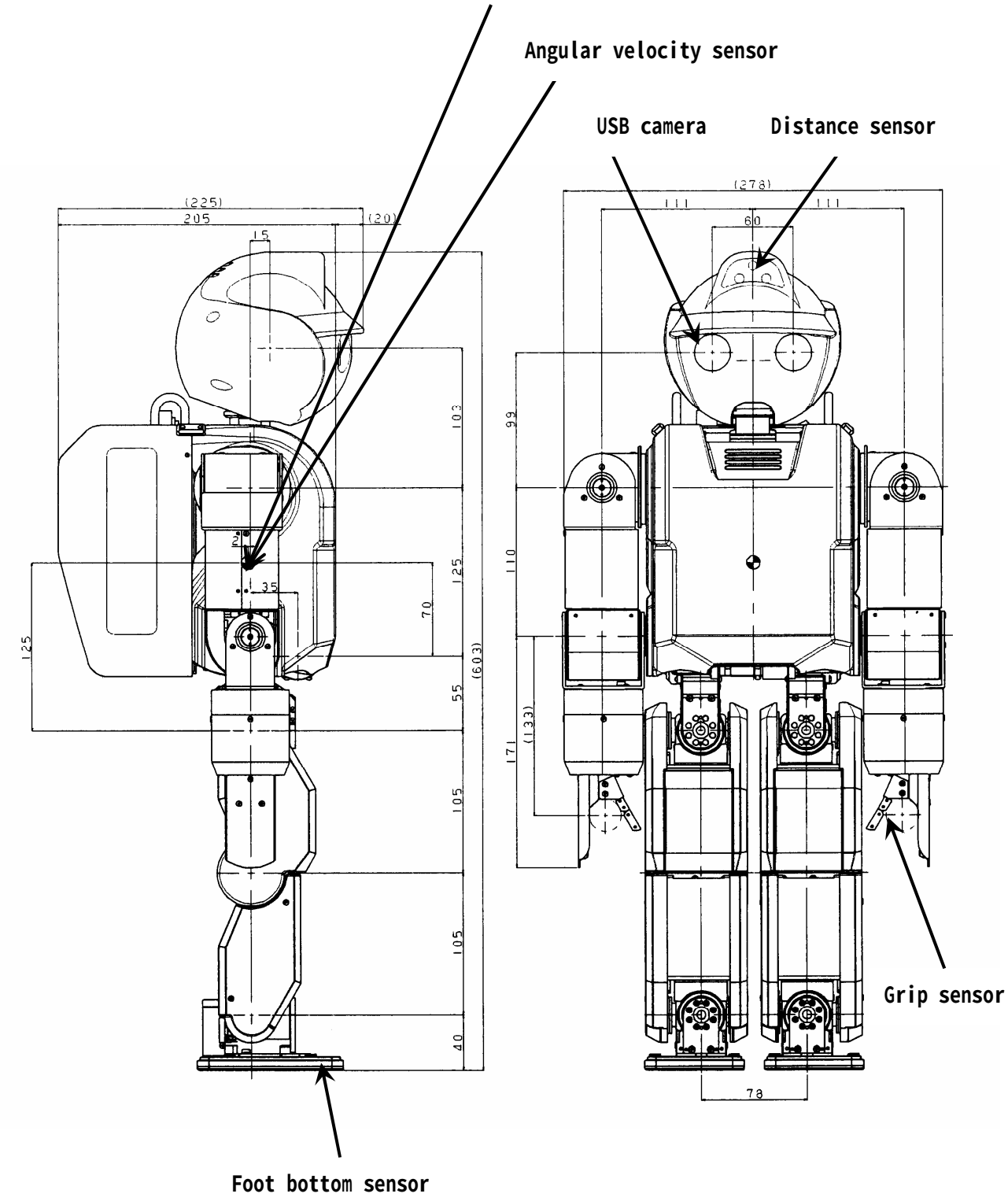


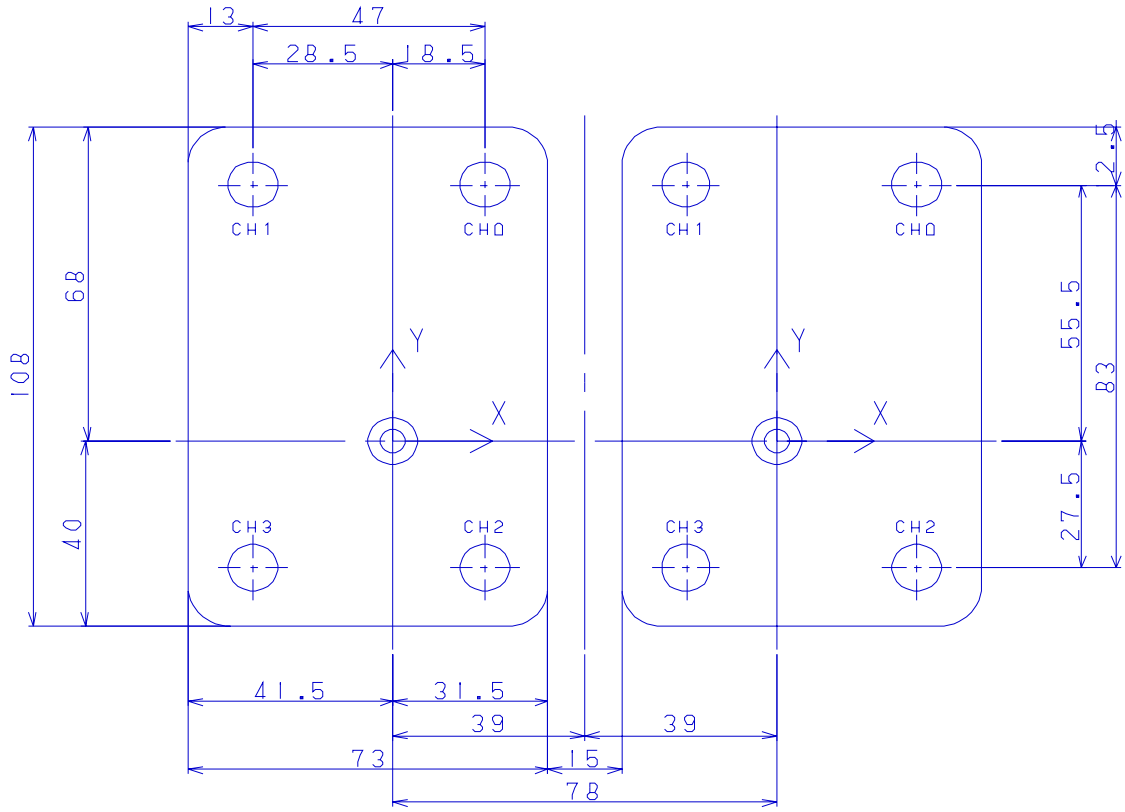
Fig.6.6.1-1 Sensor loading position

6.6.2 Posture sensor

The posture sensor is carried in the inside of the body of a robot's upper half of the body, and consists of an acceleration sensor of three axes, and an angular velocity sensor (gyroscope) of three axes. In order to secure the accuracy of a posture sensor, please acquire the standard data of the acceleration in a robot's erection stillness condition, and angular velocity, and compute the amount of physics absolutely in the amount of change from this data. The conversion type which computes the physical data in the WAIST coordinate system which is a robot's standard coordinates from the ADC result of each sensor is shown in the data of an attached sheet. Keep in mind that the order of an ADC channel number of an acceleration sensor is not X of a robot coordinate system, Y, and the order of Z with the polarity.

6.6.3 Foot bottom sensor

The tip of a foot of both the robot's legs is equipped with the Foot bottom sensor which detects the perpendicular anti-power (W) of a zero moment point (ZMP) and a floor concerning the leg at the time of carrying out BETA leg landing. The detection element carries [pair of shoes] the admiration pressure sensor (FSR element) from which resistance changes according to pressure in four pieces /, and computes ZMP and W using this resistance change. Since the characteristics of a sensor differ for every individual of a robot, it computes by the individual correction formula attached in the attached sheet. The form and ZMP detection coordinates of a Foot bottom sensor are shown in the following figure.



The ankle joint position in an erection posture

X, Y ZMP detection coordinates

Fig. 6.6-2 ZMP detection coordinates of foot botom

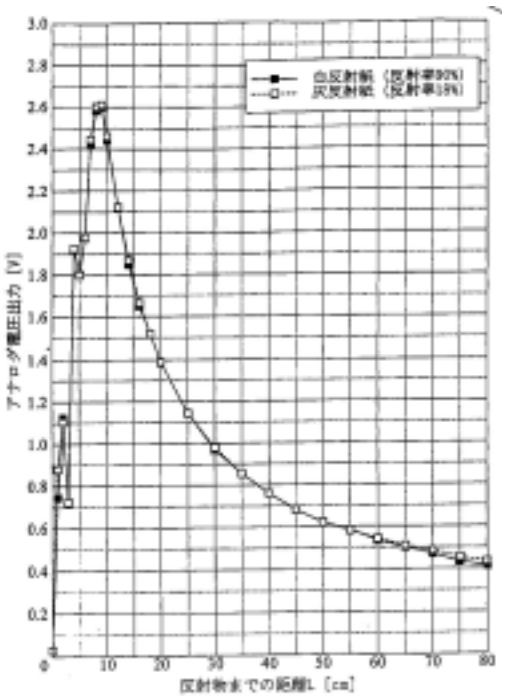
6.6.4 grip sensor

It carries the grip sensor at a time in the one thumb of both hands. Since it is attached to the finger, this sensor may change a sensor output value a lot, even when the same object is griped by the influence of the form of grip thing, the surface, etc., the direction of grip, etc. Please specify the Threshold value of grip existence etc by grip thing or its grip posture.

Fig. 6.6-3 grip sensor

6.6.5 Distance Sensor

The distance sensor is carried in the one center of a head. The characteristic of this sensor is shown in Fig. 6.6-4. The relation between the AD translation value inputted by the sensor processing board and the analog voltage output of this sensor is about called for by the following formulas.



(Sensor AD translation value [LSB])

=(analog voltage value of sensor [V]) x200

For example, when an obstacle is in the position of about 10cm, graph to an output voltage value is about 2.6V. The AD translation value at this time is set to about 520 (= 2.6x200).

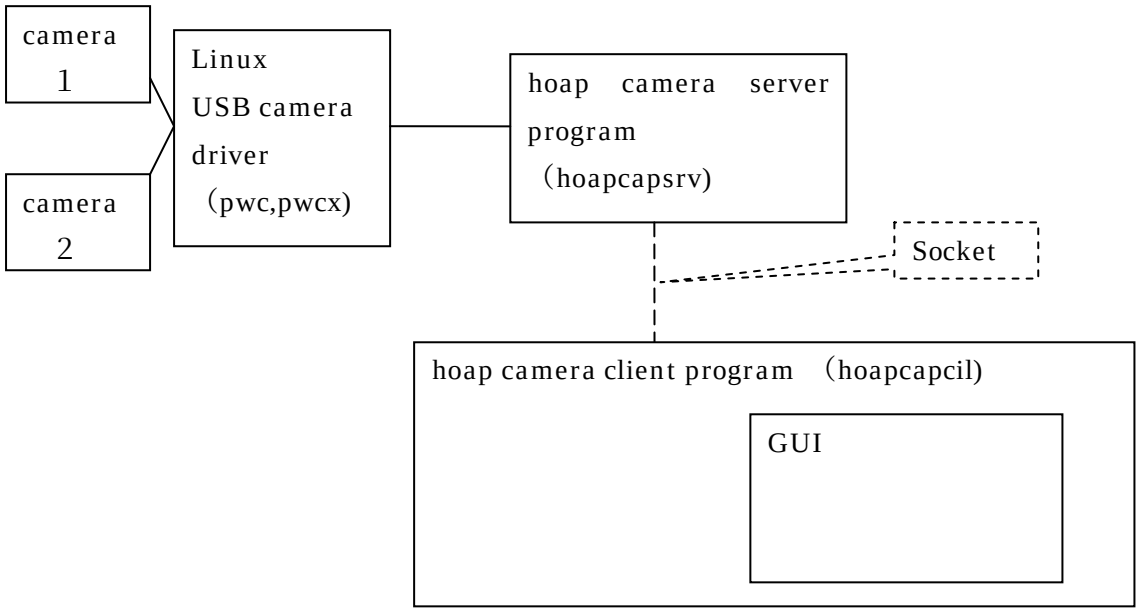
6.6.6 Camera

Cautions:

When both a USB camera and a sound device are used at the time of cable operation Please do not connect the USB terminal of a camera to Exterior PC at the time of starting. Please insert after starting. The sound card of Exterior PC will become invalid and the microphone of a camera will become effective.

Two USB cameras are attached in the head of HOAP?3. The USB camera driver will be set up in the state at the time of shipment.

(1) Camera sample operation procedure



1) Start a camera server program with PC which connected the HOAP camera.

#cd /usr/local/hoap3/hoapcapsrv-x.x.x (x.x.x is Version)

#./hoapcapsrv

2) PC (only the HOAP3 exterior PC is installed) with which X-Window was installed

#cd /usr/local/hoap3/hoapcapcil-x.x.x (x.x.x is Version)

#cd ./hoapcapcil xxx.xxx.xxx.xxx (xxx.xxx.xxx.xxx is server IP)

(In the case of self-PC, it is 127.0.0.1)

3) A camera picture is displayed on GUI of a client program.

Cautions:

Port number used for socket communication is specified by

/etc/services file of hoapserver and 9999/tcp.

Please rewrite, when change is required. (server PC / client PC)

(2) Sample image processing

A check of each check box operates image processing of a sample.

flip:Vertical reversal

gray:Monochrome processing

smooth:Smoothing

linedraw:Line drawing display

(3) Notes

1) The USB connector of a camera option should put to port where a host controller is another with the USB connector of HOAP. In the case of an example attached personal computer, in HOAP, a back port and a camera serve as a front port.

2)Please use it after removing comment out of USB_UHCI of /etc/modules.conf, in order to make UHCI usable.

3)Depending on the personal computer to be used, real-time control of 1msec may not be able to be performed at the time of camera unit use.

4)Please distinguish distinction of right and left of a camera by the user program side by USB-ID of each camera.

7. Program and Control

7.1 Local Control

As for this robot, a micro-controller is arranged to each joint, and Local High gain feedback control is done in every joint, and take the form that that movement target value is directed delicately from the motion command personal computer. Because of that, each micro-controller doesn't do an orbit calculation such as so-called trapezoid method. Control is done to meet that substitute command value promptly which a user command. USB which is the standard general-purpose interface of the personal computer is adopted for the communication between the personal computer and the micro-controller. RT-Linux is adopted as a real-time operation system in the motion command personal computer.

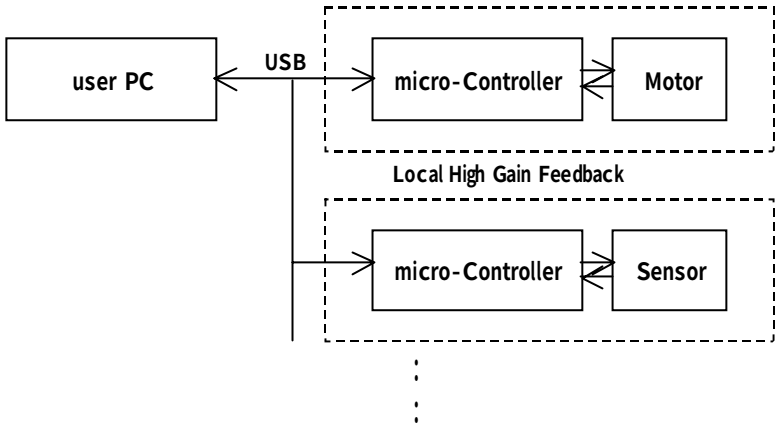


Fig. 7.1 Control composition of HOAP-3

There are that advantage and a restriction by having adopted USB in the communication form. Understand it about the character of USB when you handle this robot system.

Advantage:

- Communication speed is comparatively fast(12Mbps)
- Many devices can connect (up to127 units), communication system can be easily expanded.

Restriction:

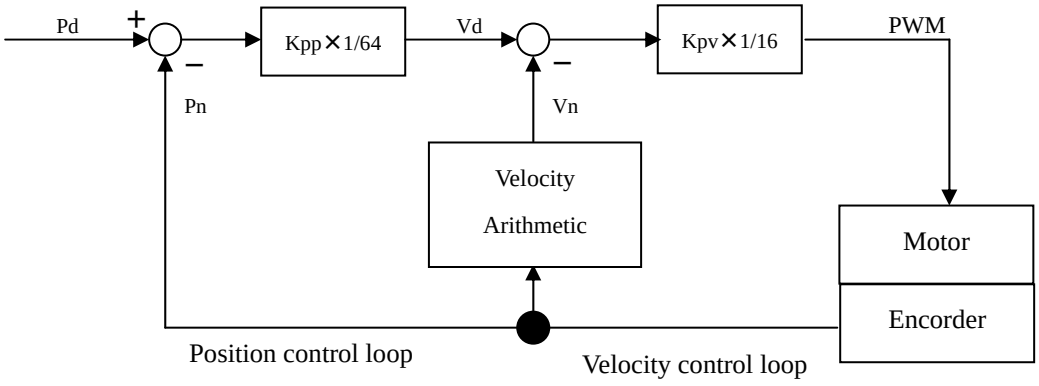
- There are a host, a general idea of host and function, and a host side controls all of the communication fundamentally. In other words, it can't communicate actively from the function side. As for this system, PC is equivalent to Host, Micro-controller is equivalent to function.
- A device for the divergence caused with HUB for the increase of the function is necessary.
- Communication method is restricted due to communicating in the frame unit of every 1mS so called SOF.

7.1.1 Motor control board & Sensor board

21 motor control boards and four sensor processing boards which control each motor required for attitude control, and two RC motor control boards are carried in the robot. With the various commands explained by 7.2.1, a setup and the present state of data are transmitted and received. Here, the firmware structure of the motor control board and a sensor processing board is explained.

7.1.1.1 Motor Control Board

The block diagram of the firmware which equips it with a motor control board is shown in the Drw 7.1.1-1. When the speed control mode is taken, Vd to say in the following block diagram is decided to be direct directly. A speed is calculated by the number of changes in a pulse of the unit 1mS from the one for the difference of the encoder.



Drw7.1.1-1. **Motor Control board block drawing**

7.1.1.2 Sensor Management Board

As for the sensor management board,6 channels of AD are being output when a running average for 8 times is taken respectively for 1mS, It begins to read the condition of the IO port which it prepares for by the option, and it is condition within 1mS just before a demand for data. (But, an IO port is connected nowhere at present.)

7.1.1.3 RC Motor board

RC Motor board is connected to RC Motors of head pan, head tilt, and both hands.

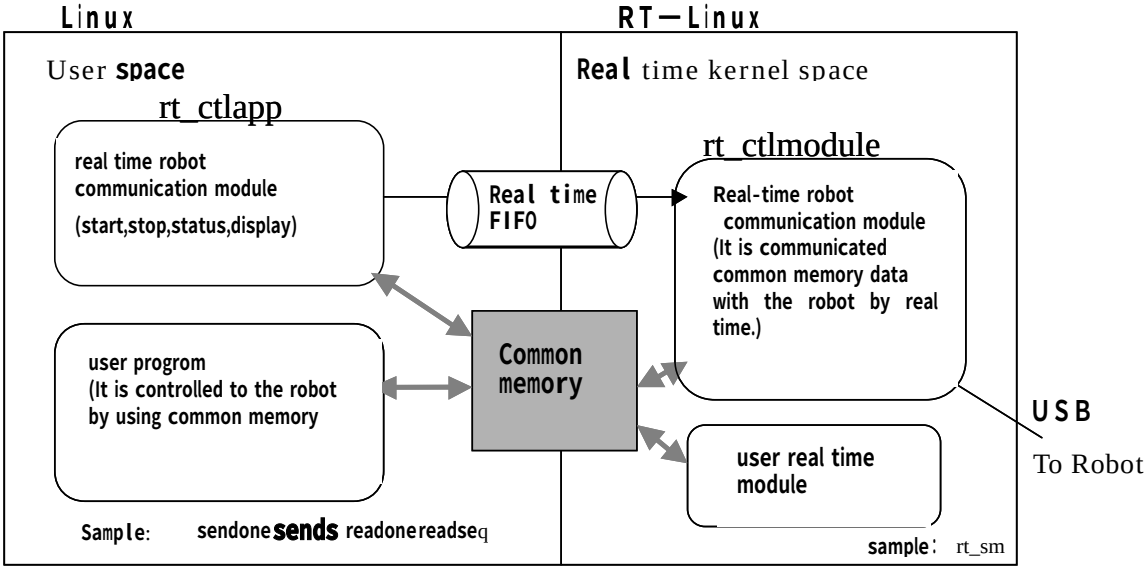
When RC motors are controled, it gives motor code and moving angle.

7.1.2 Software on Motion Command PC

Explains about the construction of the software of this robot.

7.1.2.1 Whole composition

Software on the motion command personal computer of this robot is greatly divided into the real-time robot communication module carried out with real time kernel space and the program which loading ･ indicates data on the robot carried out in the user space. A data loading, indication program and a real-time robot communication module transmit and receive data and a command by using the common memory. Common memory can writing and reading from every process. Data loading program is written data and a command in the common memory to acquire from the file and the standard input. a real-time robot communication module that content of written in the common memory is transmitted and writes the result received from the robot in the common memory. A data display program indicates the result written in the common memory.
It becomes possible that a robot is made to operate by a user's program if user make program that is reading/writing of the common memory



Drw.7.1.2-1Software composition on motion command PC

7.1.2.2 Directory composition

Software of this robot is installed in /usr/local/hoap3

/bin	¦Execute form file
/data	¦The sequence data file of the robot
/include	¦Header file
/modules	¦Real Time Module
/src	¦Source file
/sm_access	¦User Program to access in common memory
/examples	¦Sample file
/rt_sm	¦Sample of real time control mode

Explain in detail for each programs at 7.3

The sources of all programs is open except real-time robot communication module and real-time robot communication module.
Refer to it in case of individual user program preparation.

7.2 Control method of Robot

It is required to control Robot to use prepared command, and under its restriction, must do the communication of the data It prepares for the three kinds of operation modes, and it is possible that the command which shows it in each of the Drw. 7.2-1 is used as for this robot system. Understand it about the name of the mode and the command and that outline in this chapter. Explain below how to set it up referring to the structure (MemMan.h and Drw.7.2-2) of the sample which is necessary for the access actual common memory

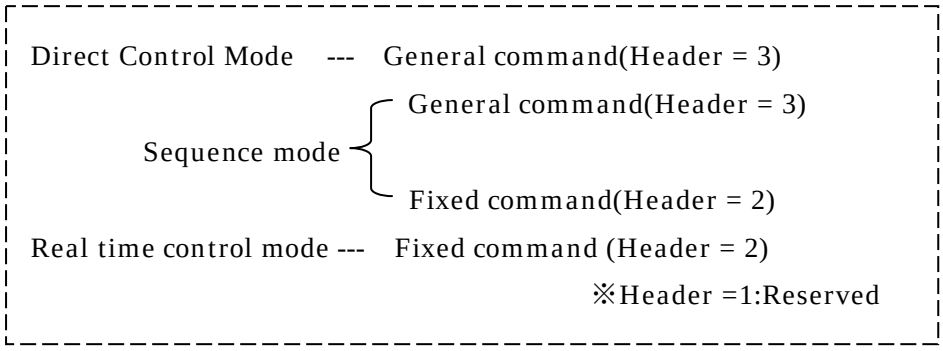


Fig.7.2-1 Relation Operation mode and Kinds of command

7.2.1 Kinds of command

Explain here for outline of commands which compose operation mode.

7.2.1.1 General Command

It is the mode which supports all the commands to a device. Each command is variable length and has the command shown in Fig.S2-1 according to a device. However, in case this command is used, it takes the time for a minimum of 2ms per one command. Namely, it becomes this thing for at least 54ms using a general command to send instructions to 27 devices.

7.2.1.2 Fixed command

It is the communicate mode of the fixed length specified for usage. Data communications can be done to all the devices (as for this basic system, 23 motor boards and 4 sensor boards) within 1mS. This velocity is applied in terms of communication when shipping.

```
typedef struct {
    USHORT   Step;    // Interval until next transmission
    UCHAR    Header;
    union {
        struct {      // Header = 2 (Fixed command)

            USHORT   MtrS[MotorMax];      ←Motor command value
            UCHAR    SnsS[SensorMax];     ←Sensor command
            USHORT   MtrRp[MotorMax];     ←Position correspondence
            USHORT   MtrRv[MotorMax];     ←Velocity correspondence
            USHORT   MtrRi[MotorMax];
            union {
                UCHAR        SnsRB[SensorMax][SnsResMax];  ←Sensor return value
                USHORT       SnsRW[SensorMax][SnsResMax/2];  (read in the size of the return value)
                ULONG        SnsRL[SensorMax][SnsResMax/4];
                LONGLONG     SnsRLL[SensorMax][SnsResMax/8];
            };
        }Ctl;
        struct {      // Header = 3  (General command)
            UCHAR    DevID;
            UCHAR    CmdAsc;
            union {
                UCHAR    ArgB[ArgMax];     ←The argument of the general command
                USHORT   ArgW[ArgMax/2];   (read in the size of the return value)
                ULONG    ArgL[ArgMax/4];
            };
            union {
                UCHAR        ResB[SnsResMax];  ←Correspondence of general command
                USHORT       ResW[SnsResMax/2];    (read in the size of the return value)
                ULONG        ResL[SnsResMax/4];
                LONGLONG     ResLL[SnsResMax/8];
            };
        }Cmd;
    };
    USHORT   ResTime;  ←Correspondence time
    UCHAR    MtrMode[MotorMax];
    UCHAR    StepOver;
}DUNIT;

typedef struct {
        :
    DEVICE   Device[nDevice];    ←The condition of the connection device. It defines separately
                                 in Usbhc.h.
        :
  }VARIF;

typedef struct {
    BOOL     UsbResetStart;      ←Reset USB(=1)
        :
    BOOL     InterruptSend;      ←Start general command(=1)
        :
    BOOL     ResRep;             ←Complete real-time control data communication
    BOOL     ResInt;             ←Complete general command data communication
        :
    int      Mode;               ←Mode settlement(1:real-time control,2:general command,
                                 combine trigger)
        :
    char     MtrMode[MotorMax];  ←Control mode of motor
        :
  }VARIF;
```

Fig.7.2-2.Explantion of MemMan.h

The complement explanation of "the Fig.7.2-2 MemMan.h". It accesses a memory with the structure to show here, and command transmission and answer data are acquired by the sample program.

Fixed Command

(1) Motor command value MrtS[]

MrtS[MotorMax]:It appears that encoder counter value in case of position command mode.

↑

Numbers of installation numbers of motor control board:when ship out :MotorMax =21

The numerical value of [] copes with order (in bottom of the order from the left column) of the table 6.1-1 with 0 - 20.

Example) MrtS[0]:RLEG_JOINT[1] The right thigh joint twisted.
MrtS[1]:RLEG_JOINT[2] The right and left of the right thigh joint
. . . . . .
MrtS[6]:RARM_JOINT[1] Front & rear the right shoulder
. . . . . .
MrtS[10]:LLEG_JOINT[1] The left thigh joint twisted
. . . . . .
MrtS[16]:LARM_JOINT[1] Front & rear the right shoulder
. . . . . .
MrtS[19]:LARM_JOINT[4] Left elbow
MrtS[20]:BODY_JOINT[1] Waist

(2) Sensor command SnsS[]

SnsS[SensorMax]:Sensor command of (S.2.1 command list)

↑

It has the loading number # kind of the sensor management board. The sample file name which the explanation of each use and that memory are being used for is shown in the following.:When shipping :SensorMax =4

For example, when R command is put, it answers the AD data IO port data of the sensor management board at one time. These answer data can lead with SnsRB[][] to mention later, SnsRW[][], SnsRL[][] and SnsRLL[][].

(Caution) S command with the argument can't be used in case of a fixed form command.

Numerical value inside [ ] copes with the following sensor with 0 - 2.

SnsS[0]:The right foot bottom sensor

SnsS[1]:The left foot bottom sensor

SnsS[2]:The posture sensor

(3) Correspondence value of motor MtrRp[], MtrRv [] , MtrRi[]

MtrRp[MotorMax]：A result of a position answer for the latest command value

MtrRv[MotorMax]：A result of a velocity answer for the latest command value

MtrRi[MotorMax]：NA (No meaning data)

The numerical value of [ ] copes with the same joint as the case of the motor command value of (1).

(4) Sensor return value SnsRB[][], SnsRW[][], SnsRL[][], SnsRLL[][]

This is union which it prepares for to read these correspondence data by the various data heads because it is written in the memory that answer data in form which varies in the command are the same.

For example, when a R command was transmitted to the sensor circuit board, it understands from "S .2.1 command list ", as for the correspondence data

The AD change data of CH0： ushort Data

The AD change data of CH1： ushort Data

The AD change data of CH2： ushort Data

The AD change data of CH3： ushort Data

The AD change data of CH4： ushort Data

The AD change data of CH5： ushort Data

The data change of IO port： uchar Data

Because the head address of SnsRB[][], SnsRW[][], SnsRL[][], SnsRLL[][] are all same,

In case of right foot bottom sensor,

The AD change data of Ch0： SnsRW [0] [0] defined in the ushort type

The AD change data of Ch1： SnsRW [0] [1] defined in the ushort type

The AD change data of Ch2： SnsRW [0] [2] defined in the ushort type

The AD change data of Ch3： SnsRW [0] [3] defined in the ushort type

The AD change data of Ch4： SnsRW [0] [4] defined in the ushort type

The AD change data of Ch5： SnsRW [0] [5] defined in the ushort type

The data of IO port： SnsRB[0][12]defined in the uchar type

The first 0 becomes 1 in case of the left foot bottom sensor.

In case of Posture sensor

The AD change data of Ch0： SnsRW [2] [0] defined in the ushort type

The AD change data of Ch1： SnsRW [2] [1] defined in the ushort type

The AD change data of Ch2： SnsRW [2] [2] defined in the ushort type

The AD change data of Ch3： SnsRW [2] [3] defined in the ushort type

The AD change data of Ch4： SnsRW [2] [4] defined in the ushort type

The AD change data of Ch5： SnsRW [2] [5] defined in the ushort type

The data of IO port： SnsRB[02[12]defined in the uchar type

The interpretation of each data should consult description of the sensor characteristic of an attached sheet and "6.6.2 posture sensor", a "6.6.3-bottom sensor", and a "6.6.4-grip & distance sensor." In addition, CH.4 of a right leg bottom sensor acquire the voltage data of a battery as indicated by "Fig.S1.4 battery exchange method." When set to 550 or less LSB, please remove a battery promptly.

General command

(1) No. of transmitted device DevID

It is the number of the device which becomes a place of transmission of a general command to transmit only to the specified device. Defined as below.

1: RLEG_JOINT[1] Motor control board of joint(capture 6.1-1reference)
2: RLEG_JOINT[2] Motor control board of joint
3: RLEG_JOINT[3] Motor control board of joint
4: RLEG_JOINT[4]Motor control board of joint
5: RLEG_JOINT[5] Motor control board of joint
6: RLEG_JOINT[6] Motor control board of joint
7: RARM_JOINT[1] Motor control board of joint
8: RARM_JOINT[2] Motor control board of joint
9: RARM_JOINT[3] Motor control board of joint
10: RARM_JOINT[4] Motor control board of joint
11: LLEG_JOINT[1] Motor control board of joint
12: LLEG_JOINT[2] Motor control board of joint
13: LLEG_JOINT[3] Motor control board of joint
14: LLEG_JOINT[4] Motor control board of joint
15: LLEG_JOINT[5] Motor control board of joint
16: LLEG_JOINT[6] Motor control board of joint
17: LARM_JOINT[1] Motor control board of joint
18: LARM_JOINT[2] Motor control board of joint
19: LARM_JOINT[3] Motor control board of joint
20: LARM_JOINT[4] Motor control board of joint
21: BODY_JOINT[1] Motor control board of joint
22: RC motor control board
(HEAD_JOINT[1], HEAD_JOINT[2], HEAD_JOINT[3])
23: RC motor control board
(RARM_JOINT[5], RARM_JOINT[6], LARM_JOINT[5], LARM_JOINT[6])
24: The right foot bottom treatment board
25: The left foot bottom treatment board
26: Posture sensor treatment board
27: grip & distans sensor treatment board

(2) Command　CmdAsc

They are command to transmit each device. Refer to「S.2.1Command list table」

(3) Command reference　ArgB[], ArgW[], ArgL[]

It is reference data to transmit to above command. Refer to「S.2.1 Command List table」

It is provided different arrangement of data length union so that the data format of the various commands may respond as well as the sensor return value.

(4) Response of General Command　ResB[], ResW[], ResL[]

It is response data from devices that transmitted. This is also declared the arrangement of the different data length with union to be able to comply with the various data formats in the same way.

(5) Response time　ResTime

It is response time result from actual devices.

(6) Motor Control Mode　MtrMode[]

They are the data which show the control mode of the motor set up.

Refer to [S2.1 Command list table]

(7) Step Over　StepOver

Not used

VARIF Structure

```
typedef struct {                        // Driver   User
        BOOL    UpStreamEnable;         // W        R       // It can't use.
        BOOL    UsbResetStart;          // R(W)     W       // USB bus reset transmit start
        int     ConfiguredHB;           // W        R       // setups complete HUB number
        int     ConfiguredDD;           // W        R       // setups complete device No.
        int     ConfiguredIDX;          // W        R       // setup complete index
        BOOL    ConfiguredSS;           // W        R       // It can't use
        BOOL    ConfiguredCD;           // W        R       // It can't use
        BOOL    DAction;                // W        R       // It can't use
        BOOL    SendStart;              // W        R       // It can't use
        BOOL    SendOK;                 // W        R       // It can't use
        BOOL    InterruptSend;          // R(W)     W       // The transfer start command of the general
                                                                    command
        BOOL    Received;               // W        R(W)    // Response receiving completion
        BOOL    ResRep;                 // W        R(W)    //  〃  for fixed form command
        BOOL    ResInt;                 // W        R(W)    //  〃  for general command
        USBERR  UsbErrMon;              // W        R(W)    // Transfer status
        USBERR  UsbErrUs;               // W        R(W)    // It can't use
        int     Mode;                   // RW       W       // Transfer mode command/monitor
        int     SendLength;// W         R       // Command size (Referenced)
        int     ResLength;              // W        R       // Response size (Referenced)
        int     cRes;                   // W        R       // Response receiving counter
        int     ReceiveLength;          // W        R(W)    // Response size (Actual)
        int     ResTime;                // W        R       // Response time
        int     ResTimeMin;             // W        RW      // Minimum response time
        int     ResTimeMax;             // W        RW      // Max. response time
        double  ResTimeAvr;             // W        RW      // It can't use
        char    MtrMode[MotorMax];      // W        R       // Current each motor mode
        BOOL    UpStreamSendStart;      // R(W)     W       // It can't use
        int     DBG;                                     // It can't use
}VARIF;
```

Explains to the one by the function group related to the command transfer about the useful thing.

(1) USB Reset

UsbResetStart **Read/Write**

It specially explains to the one by the function group about the command transfer for the useful thing. When a user sets TRUE, a USB bus reset sending out starts. It becomes FALSE at once after a driver completes acceptance. Emulate management of all the USB devices which contain HUB is started.

(2) Related to vice emulation

ConfiguredHB **ReadOnly**

Store HUB number that finished emulation.

ConfiguredDD **ReadOnly**

Sore device number that finish emulation.

ConfiguredIDX **ReadOnly**

Shows index of HUB or Device that completed emulation imminent.

(a)General command sending

InterruptSend **Read/Write**

When the general command transmitted by transfer start command (It is effective during the idol or only the real-time control mode.), INTERRUPT direct control mode shouldn't set up. Instead of it, set up InterruptSend = TRUE。A driver becomes InterruptSend = FALSE when this flag is received. At this time, INTERRUPT is set up in VarIF.Mode inside the driver. This is because it could handle a general command as an interruption transfer in the real-time control mode.

(b)Transfer complete flag

A user must reset each transfer complete flag in FALSE before the command transmission. These flags become TRUE when a transfer is completed. Though "transfer completion" to say here is "command transmission - a response receipt", it becomes TRUE at the moment when command transmission was completed at the case of the general command in case of the one without a response.

in case of the real-time control mode, renew command value, after Received or ResRep was made FALSET,when it confirmed that that flag became TRUE, renewed command value received by target, or response value becomes to renewal. Usually, it is possible to confirm transfer completion by using Received, but you must confirm each transfer completion by using ResRep / ResInt when an interruption transfer in general command form is specially done in the real-time control mode

Received **Read/Write**

Transfer completion (common)

ResRep **Read/Write**

Transfer completion (for fixed form command)

ResInt **Read/Write**

Transfer completion (for general command)

(c) **USB** transfer status

UsbErrMon **Read/Write**

It is constructed by the structure USBERR. cBitStuffErr - cStalled is counter which counts information such as abnormal in USB protocol that it is formed by USB Host Controller . cResTimeOut and TDEndTO is the information that it is formed by driver . A user can reset counter by writing 0.

```
typedef struct {                    // Driver User
    int     cBitStuffErr;   // W    R(W)
    int     cCRCTOErr;              // W    R(W)
    int     cNAKReceiveOut;         // W    R(W)
    int     cNAKReceiveIn;  // W    R(W)
    int     cBBLDetect;     // W    R(W)
    int     cDBuffErr;              // W    R(W)
    int     cStalled;               // W    R(W)
    int     cResTimeOut;    // W    R(W)   // Response receiver time out.
    BOOL    TDEndTO;                // W    R(W)   // bucket sending time out.
}USBERR;
```

(d) Transfer mode command/monitor

Mode **Read/Write**

Set up value of transfer mode fixed number definition ModeConstants. A driver does the movement to be equivalent to each mode. But, it is only IDLE, REPITITION and SEQUENTIAL that a user can set. Refer to InterruptSend in case of general command transfer.

```
enum ModeConstants{
  IDLE,                 // Idle
      REPITITION,       // Real time comtrol
  SEQUENTIAL,           // Sequence
  INTERRUPT,            // Direct control (user cannot set up)
UPSTREAM                // It can't use
  };
```

(e)Transfer Data size

SendLength **ReadOnly**

Transmit command size (Referenced)

The command size defined inside the driver in advance for every transmitting command is stored.

ResLength **ReadOnly**

Response size (Referenced)

The response size defined inside the drive in advance for every transmitting command is stored.

CRes **ReadOnly**

It is Response receiving counter to use inside of driver.

Become receipt completion at cRes == ResLength

ReceiveLength **Read/Write**

Response size (Actual)

It is renewed when a response receipt is completed in the size that it was actually received.

(f)Response time

ResTime **ReadOnly**

Response time

When a transfer is started, initialization (0) is done by a driver. After a command is transmitted, transfer time to the response receipt is set in the frame (1mS) unit. On the other hand, response time ResTime exists in member of the structure DUNIT as well. The value of each ResTime of VARIF and DUNIT is shown by each condition.

A pointer is made pDunit to the DUNIT structure of the object.

1. In case receive normal response,
   VarIF.ResTime = n
   pDunit->ResTime = n
   n : Flame No. used actually.([ms])
2. In case of transmitting general command with no response.
   VarIF.ResTime = 0
   pDunit->ResTime = 1
3. When an invalid device ID was specified with a general command.
   VarIF.ResTime = No renewal
   pDunit->ResTime = -1

※ A command isn't transmitted and finished. Be careful in the direct control mode because a transfer complete flag isn't set, either. But, a step progresses in case of the sequence mode.

※ When it was abnormal finished by the host controller.

VarIF.ResTime = -1

pDunit->ResTime = -1

The contents of the abnormal can be confirmed by using counter of UsbErrMon.

4. When a time-out (1sec) was done by the NAK receiving continuation and so on.

,VarIF.ResTime = 0

pDunit->ResTime = not renewal

The occurrence of the time-out can be confirmed by using counter of UsbErrMon.

The minimum value of ResTime and maximum value are stored in ResTimeMin/ResTimeMax.。 A user can initialize it by writing 0 respectively.

ResTimeMin **Read/Write**

Minimum response time

ResTimeMax **Read/Write**

Maximum response time

(g) Motor control Mode

MtrMode[MotorMax] **ReadOnly**

Each present motor control mode is stored, and [device ID - 1] of the motor can be referred to as an index. The number of the motor control mode change command transmitted at the end is stored.

('A'/'B'/'C' = Position / Velocity / <reservation>)

Refer to this for this information except for the sequence mode though it is the same as the thing of the structure DUNT inside usually. MtrMode inside DUNIT is effective when the motor control mode is confirmed in each step which has been carried out in case of the sequence mode.

The state of connection condition when ship out of a sensor treatment circuit board as following.

table Connection condition of sensor treatment board when ship out

Type	Right foot bottom Sensor treatment board	Left foot bottom Sensor treatment board	Posture sensor Treatment board	Reference
Fixed command arrangement	0	1	2	[]of SnsS[] value
General command DevID	22	23	24	
AD0	FSR sensor ch0	FSR sensor ch0	Acceleration (z)	Robot coordinate xyz
AD1	FSR sensor ch1	FSR sensor ch1	Acceleration (y)	Robot coordinate xyz
AD2	FSR sensor ch2	FSR sensor ch2	Acceleration (x)	Robot coordinate xyz
AD3	FSR sensor ch3	FSR sensor ch3	Gyro (x)	Robot coordinate xyz
AD4	Battery voltage monitor	Empty	Gyro (y)	Robot coordinate xyz
AD5	Empty	Empty	Gyro (z)	Robot coordinate xyz
IO7	Empty	Empty	Empty	
IO6	Empty	Empty	Empty	
IO5	Reserved	Reserved	Reserved	
IO4	Empty	Empty	Empty	
IO3	Empty	Empty	Empty	
IO2	Empty	Empty	Empty	
IO1	Empty	Empty	Empty	
IO0	Empty	Empty	The output for LED of the breast	

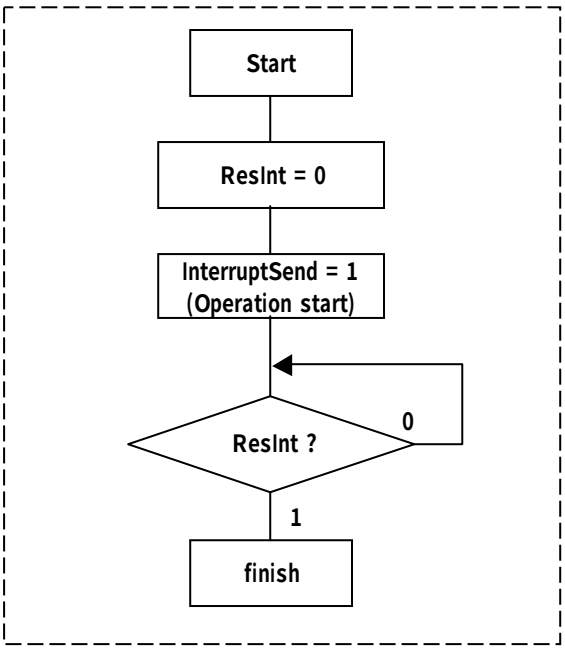
7.2.2 Kinds of operation mode

The outline is explained about the operation mode corresponding to the user's use situation. Each operation mode starts a movement by establishing the value of Mode. The correspondence of the operation mode and that value is as following.

Operation mode	Mode value	Motion start Trigger
Direct control mode	0(= IDLE)	InterruptSend = 1 set
Sequence mode	1	Mode = 1 set
Real time control mode	2	Mode = 2 set

7.2.2.1Direct control mode

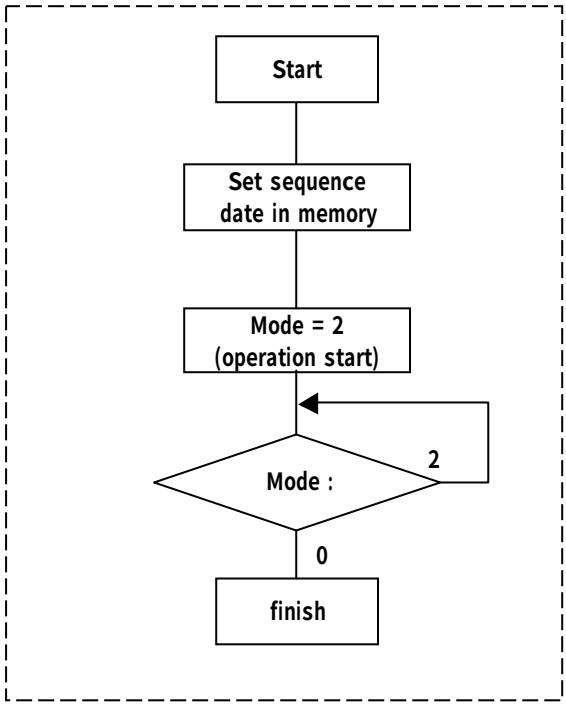
It is the mode to use when general operation is done in the device unit. A specified command is carried out in this mode on every 1 line. Only a general command can be directed to the command. Surely, the next data are set after it confirms that a general command data communications complete flag (ResInt) stands. You must stand an interrupt flag (InterruptSend) more to transmit a message actually. A flow chart in this mode is shown in the Drw 7.2-3.



Drw. 7.2-3. Direct control mode

7.2.2.2 Sequence mode

It uses when it wants to do operation continuously without manual input making a command process with macrocosm in the direct control mode . A command is enumerated in the file, and a command is carried out in the time interval specified as turn by setting it on the memory. It is possible that a fixed form command is made to be mixed with the general command in this mode. But, the processing of 1 line takes the time of at least 2mS at the general command and at least 1mS at the fixed command. It is required to stand a sequence mode start flag to carry out the mode more.

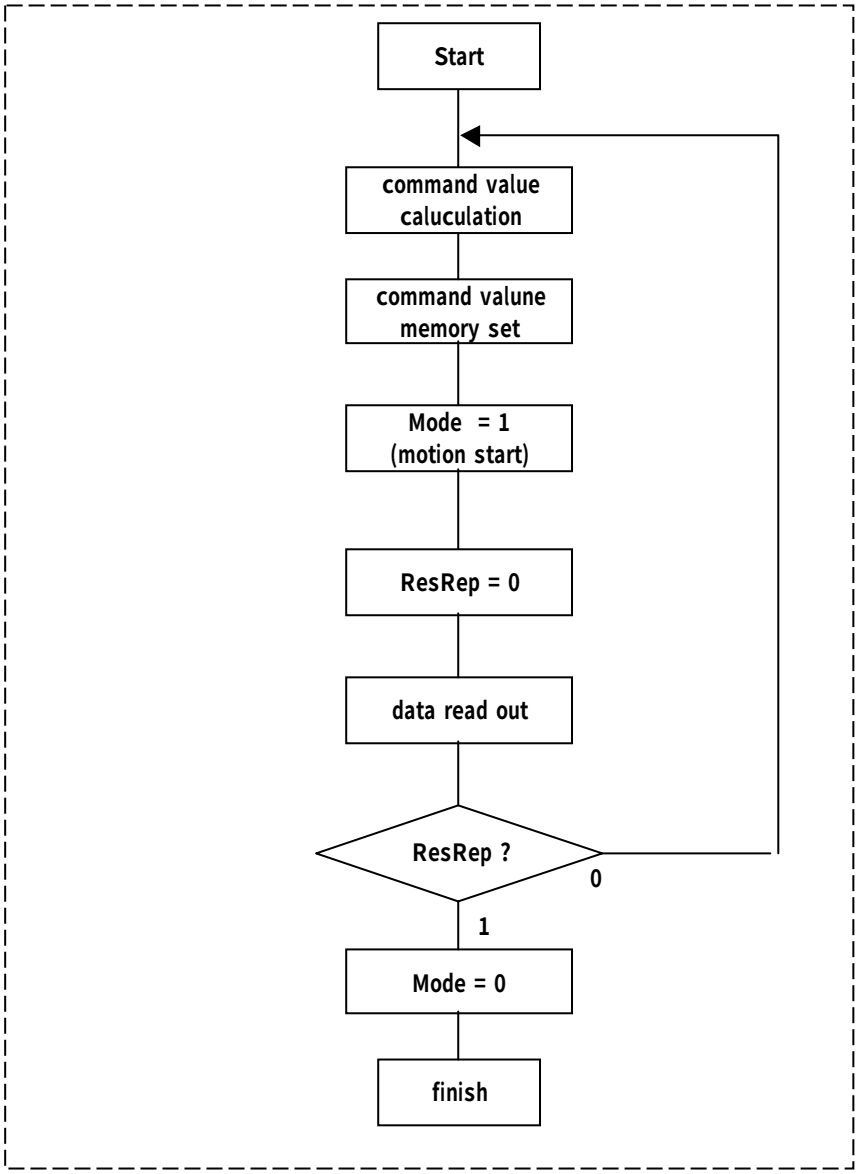


Drw. 7.2-4 Sequence mode

7.2.2.3 Real time control mode

It is the mode to use for a user when to do real-time operation such as closed loop control. A user must control with seeing a fixed form command data communications complete flag exactly the matter whether there was an effective answer or Command value can be written in .

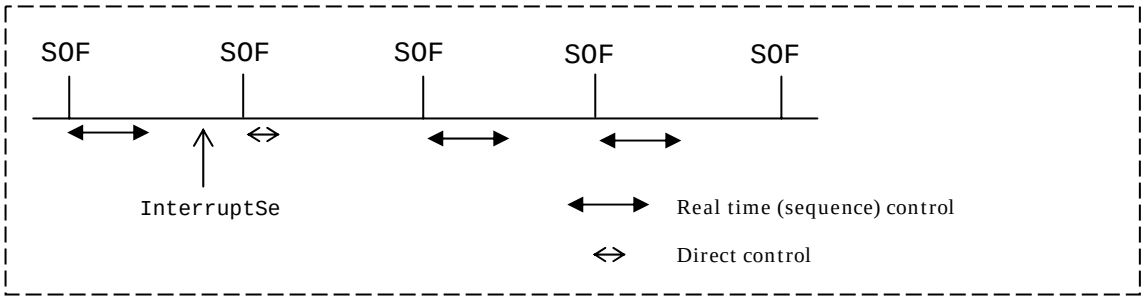
It can be controlled within 1mS if the next command value is written within 100us after command value is written in and a fixed form command data communications complete flag stands. And, you must stand a real-time control start flag to make this mode effective.



Drw. 7.2-5 Real time control mode flow chart

7.2.2.4 Mixture of Operation mode

While continuance in the real-time control mode or the sequence mode works, there is time which it wants to operate by the case to the specific device. In this case, interruption operation can be done by the direct control mode. The direct mode isn't started movement by Actually, data are transmitted with the next frame that stands an InterruptSend flag, and it is returned from the next frame again for the movement of the usual control mode. the settlement of the value of Mode, but it is to start transmission by standing a flag on InterruptSend.



Drw.7.2-6.Mixture of Operation mode

7.3 Explanation of sample soft ware

Explain software for robot on motion command PC

rt_ctlapp	
explanation	It communicates with the real-time robot communication module rt_ctlmodule.o, and it is the program which gets the practice of the start/stop of the real-time robot communication module and information on the robot.
How to use	。A RT-linux module and a robot control real-time module rt_ctlmodule.o are started by entering command as rtlinux start rt_ctlmodule.o and a command. Next, when rt_ctlapp is carried out, it becomes the prompt of *.*.*$. (*is number) First * is in the number of USB hubs, the second * is in the number of USB controller, their * indicate USB controller in the recognition. The following command can be input. r : USB reset v : Common memory status indication h : Help d : i : Direct control mode monitor r : Real time control mode monitor 2 : Setting to real time control mode 3 : Setting to direct control mode
Caution	It cannot carry out if rtlinux module is not operated. rCarry out by root.
Binary	/usr/local/hoap3/bin/rt_ctlapp
Source	Non

Sendone	
Explanation	It is the program which transmits a general command to the robot. A command is written in the common memory. A general command is carried out by transmitting the contents that a real-time robot communication module was written in the common memory to the robot itself.
How to use	Carry out from command line、 Shows prompt as command: A general command is written in the common memory when a command is written and a return. When the practice of the command is completed, return value is written down in the standard output. It is indicated continuously as command : a command can be input in succession. if practice isn't completed like a robot isn't connected , return to command: prompt by pushing any key. Finish if input with quit. Input command is <sec>,<header>,<devID>,<command>,<param1>. <sec>：Whatever is input, it doesn't care in the general command mode because it is meaningless. <header>：Input "3" which shows a general command <devID>：It is specified which USB controller of the robot a command is sent to. <command>：Input to refer to general command list <param1>..：It is a parameter to different from <command>。Input to refer to general command list. For example command: 2,3,4,J is to send general command as J in device 4.
Caution	It can't be carried out if a rtlinux module doesn't start it. Carry out by root
Binary	/usr/local/hoap3/bin/sendone
Source	/usr/local/hoap3/src/sm_access/sendone.c, sm_access.c

Sendseq	
Explanation	It is a program to send sequence data for robot. Write sequence data in common memory. A sequence motion is carried out by transmitting the contents that a real-time robot communication module was written in the common memory to the robot itself.
How to use	From command line sendseq < <filename> as, If input file name by re-direct, that file is loaded, and written in as the sequence data, and the practice of the sequence motion is indicated. If input file name by re-direct, that file is loaded, and written in as the sequence data, and the practice of the sequence motion is indicated.
Caution	It can't be carried out if a rtlinux module doesn't start it. Carry out by root
Binary	/usr/local/hoap3/bin/sendseq
Source	/usr/local/hoap3/src/sm_access/sendseq.c, sm_access.c

Readone	
Explanation	It is the program which indicates all the main things of the contents being written in the common memory at present. Contents of command transmission and that result can be confirmed.
How to use	From command line Input as readone it is indicated content of data of common memory at present.
Caution	It can't be carried out if a rtlinux module doesn't start it. Carry out by root
Binary	/usr/local/hoap3/bin/readone
Source	/usr/local/hoap3/src/sm_access/readone.c, sm_access.c

Readseq	
Explanation	While a sequence motion is carried out, it is a program that reads a result to the place where a movement was finished one after another from the common memory and to indicate. Present contents of command transmission and that result can be confirmed. It is the program which indicates the contents being written in the common memory.
How to use	From command list Input as readseq Contents of transmission to the sequence movement being carried out at present and a result are indicated. On and after, it is indicated every time the next movement during the sequence movement is finished. Finish if input anything by keyboard.
Caution	It can't be carried out if a rtlinux module doesn't start it. Carry out by root
Binary	/usr/local/hoap3/bin/readseq
Source	/usr/local/hoap3/src/sm_access/readseq.c, sm_access.c

rt_sm	
Explanation	It is the sample program which writes motion data in the common memory which uses a real-time robot communication module from the real-time module.
How to use	From command line Input as rtlinux start rt_ctlmodule.o rt_sm_module.o Start sample real time module together with real-time robot communication module. Start rt_sm_app 、sample real time module start, and data is written in common memory. rt_sm_app stops a sample real-time module, and finished when motor encoder value is received from the sample real-time module 100 times. The contents being written in the common memory can be confirmed by the readone program.
Caution	It can't be carried out if a rtlinux module doesn't start it. Carry out by root. A robot doesn't work with this source because it is a sample to write in the common memory from the real-time module.
Binary	/usr/local/hoap3/examples/rt_sm/rt_sm_app, rt_sm_module.o
Source	/usr/local/hoap3/examples/rt_sm/rt_sm_app.c

Setpos	
Explanation	Current value is made to correspond to the target command value of the robot
How to use	Input as" setpos" from command line
Caution	It can't be carried out if a rtlinux module doesn't start it. Carry out by root.
Binary	/usr/local/hoap3/bin/setpos
Source	/usr/local/hoap3/src/sm_access/setpos.c

Interpol	
Explanation	It is the program which makes the sequence data file that interpolate the space to the initial posture of the sequence data file which will be read next from the present posture of the robot.
How to use	Interpolate sequence data file name is " outputfile"、sequence file name which will carry out next is "nextfile", then, input " interpol outputfile < nextfile" from command line Make sequence data file which interpolates with two seconds to the current position specified with a head of next file from the current position of the motor of the robot. outputfile change the file of the name of interpol.csv when it is omitted.
Caution	It can't be carried out if a rtlinux module doesn't start it. Carry out by root.
Binary	/usr/local/hoap3/bin/Interpol
Source	/usr/local/hoap3/src/sm_access/interpol.c, sm_access.c

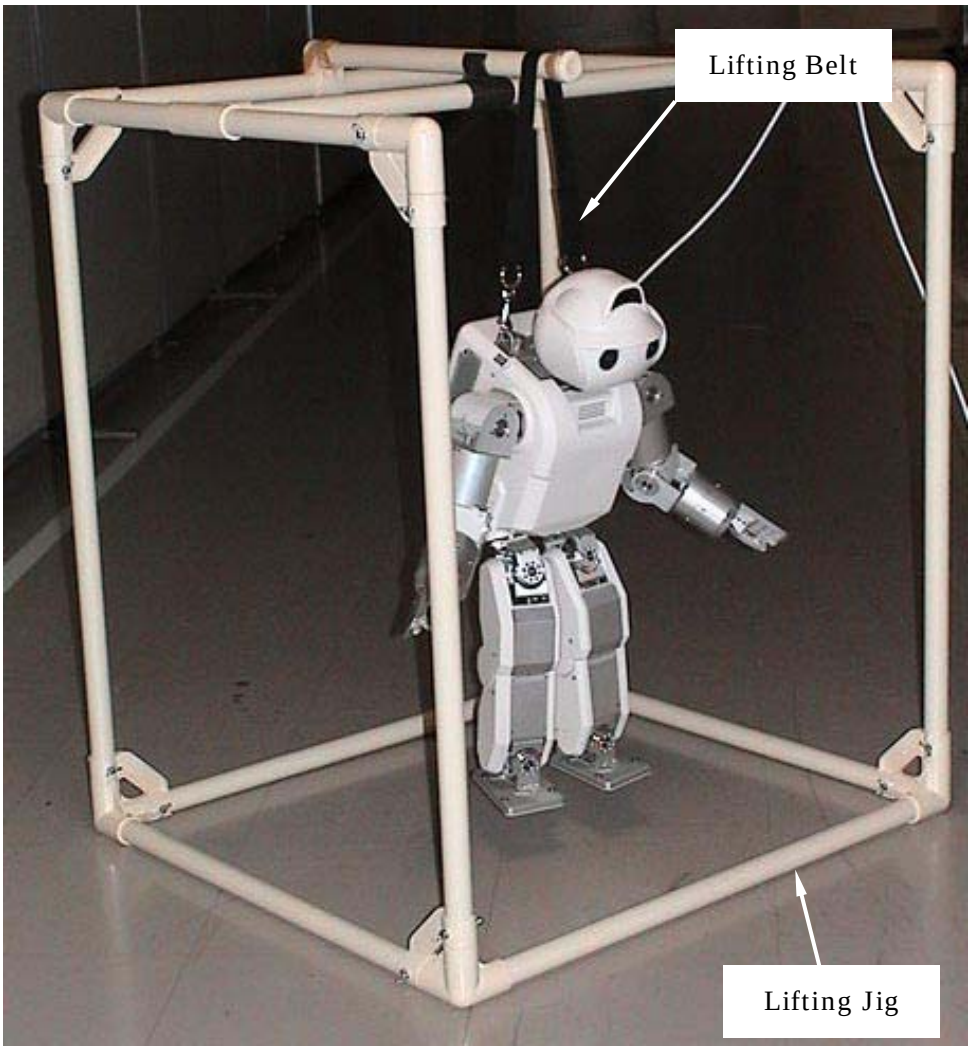
S.1 Mechanical

S.1.1 Lifting jig

Lifting jig is provided to lift robot temporary as a accessory . The purpose of jig is : When a robot is kept without giving a load to the joint under the condition that it is hung in the air、or when a joint is controlled tentatively, or when It uses for the case that the output of the sensor is confirmed, the exchange work of the part, and so on.

It passes into picked up through attached lifting belt in the top (the back of the head) of the back of the robot, and it is hung the opposite side at the tip of the pipe in the center of lifting jig, and a lifting belt.

(*) Be careful not to fall the robot to fall by the mistake when fixed the chain and so on.



S1.2 The connector joint method

This chapter is explanation about combination of the connector inside a back cover.

(1)How to remove a back cover

In order to attach and remove the connector inside a back cover, it is necessary to remove a back cover.

a) Please cancel the stop implement shown in an attached Fig S1.2-1 (it is made to pull up and rotate), and remove a front cover.

Fig S1.2-1 How to open a front cover

b) A back cover is being fixed with four screws as it is shown in a Fig S1.2-2. Please remove the four screw with a driver.

a) The side of a back cover b) The bottom of a back cover

Fig S1.2-2 How to remove a back cover

(2) Robot USB port

A robot USB port is the connector of Fig S1.2-3 in a back cover. In the case of cable mode, the USB cable from Outside PC is connected to this connector, and, in the case of radio mode, the USB cable from Inside PC is connected.

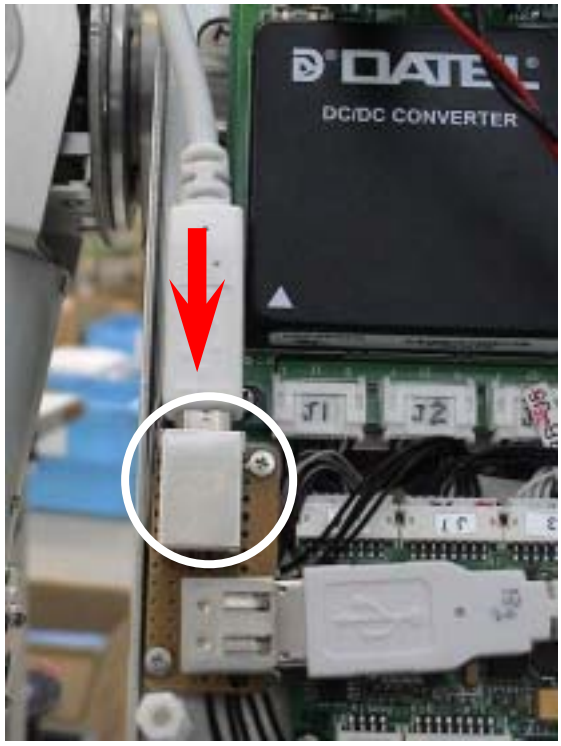


Fig S1.2-3 Robot USB port

(3) How to attach a back cover

Attachment of a back cover is performed in the order of reverse of "how to remove (1) back cover."

(4) External power supply input port

The power supply connector from an external power supply is connected to the external power supply input port (Fig S1.2-4) located on the robot upper surface at the time of cable mode.

FigS1.2-4 External power supply input port

S1.3 The battery exchange method

This chapter is explanation about the exchange method of the battery currently installed in the front cover (Fig. 1.3-2). In use, be careful below in battery mode.

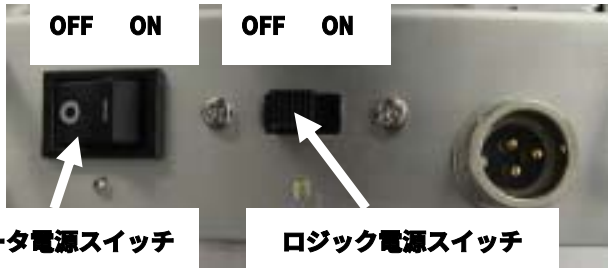
notes 1) Battery residual quantity can be supervised by CH.4 [of a right leg sensor] of A/D conversion machine CH.0?5. If this conversion result becomes below 550LSB (an equivalent for 20V), please stop an experiment promptly and perform battery exchange. Moreover, please be sure to turn OFF at the order of a motor power switch and a logic power switch before performing battery exchange.

notes 2) Battery residual quantity can be supervised by CH.4 [of a right leg sensor] of A/D conversion machine CH.0?5. If this conversion result becomes below 550LSB (an equivalent for 20V), please stop an experiment promptly and perform battery exchange. Moreover, please be sure to turn OFF at the order of a motor power switch and a logic power switch before performing battery exchange.

notes 3) Be sure to use it in the case of use of a battery after making it full charge.

（1）Cutting of a power switch

A power supply is turned OFF at the order of a motor power switch and a logic power switch (Fig S1.3-1).



付図 1.3-1 モータ電源スイッチ、ロジック電源スイッチ

（2）Removal of a front cover

Since the battery is installed in the front cover, it needs to remove a front cover for battery exchange. Refer to "Fig S1.2-1 How to open a front cover" for how to remove a front cover.

（3）Exchange of a battery

The following procedure performs removal of a battery.

- Remove the band which is fixing the battery. (Fig S1.3-2)

Fig S1.3-2 Battery fixed band

- Remove a battery connector (Fig S1.3-3).

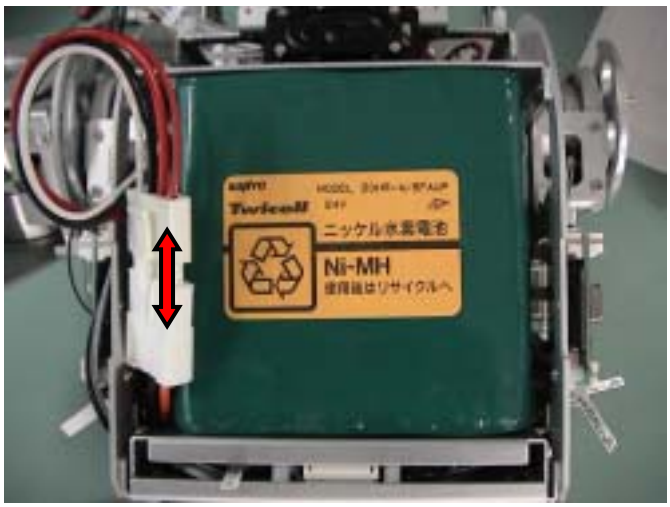


Fig.S1.3-3 Battery connector

- Finally extract a battery.

The reverse procedure of the above battery removal performs attachment of a battery.

S1.4 The timing belt adjustment meth

This robot uses two timing belts per piece. The tension of a timing belt may loosen by use of the case where excessive load is applied on structure, or a repetition. In that case, it is necessary to raise a tension again. A phillips screwdriver (tip form #0 No. and #1 No. each One .) and a 6 angle wrench are required for work.

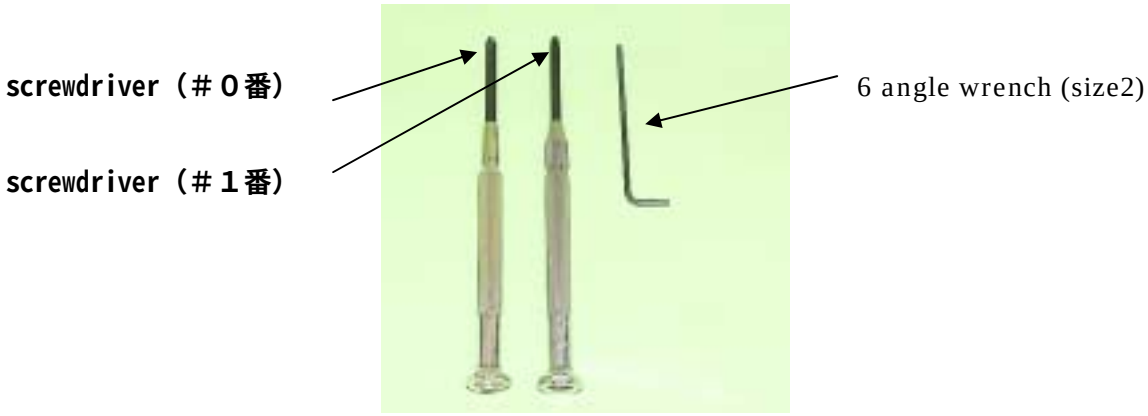


Fig S1.4-1 Tool for work

[ｓｔｅｐ1]

Two fixed screws of a Fig S1.4-2 are removed with # No. 0 screwdriver.

[ｓｔｅｐ2]

Two fixed screws of a Fig S1.4-3 are removed with # No. 0 screwdriver.

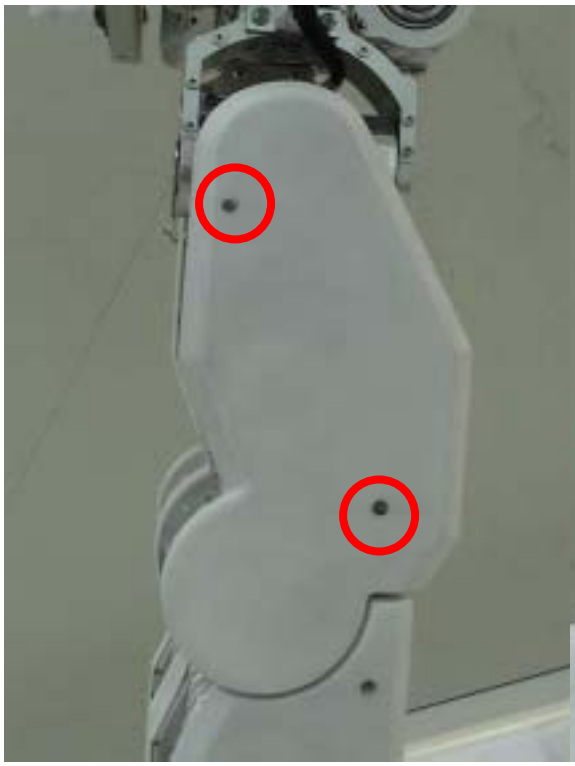


Fig S1.4-2 Leg cover fixed position

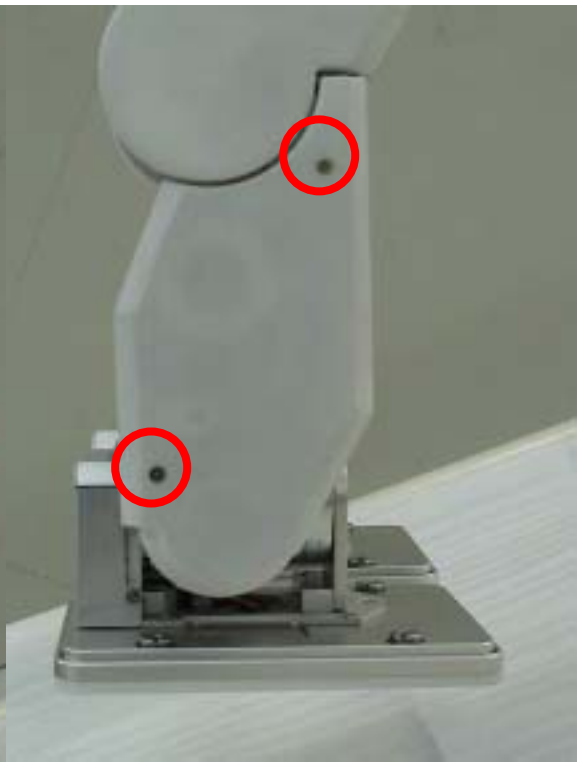


Fig S1.4-3 Leg cover fixed position

［ｓｔｅｐ３］

About 90 degrees of four screws of a Fig S1.4-4 are counterclockwise turned with a screwdriver, and a screw is loosened.

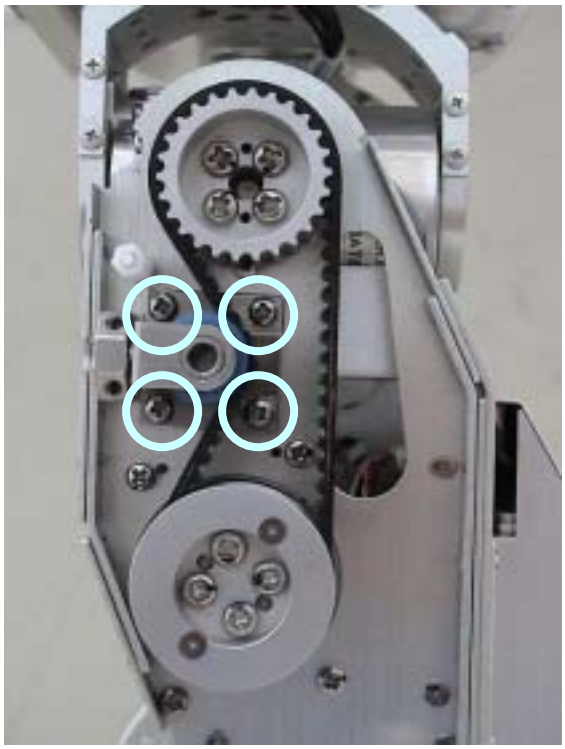


Fig S1.4-4 Adjustment part of a timing belt

［ｓｔｅｐ４］

As shown in a Fig S1.4-5, by the 6 angle wrench, please half--rotation(180 degrees)-turn the adjustment screw of a belt adjustment part clockwise, and adjust it.

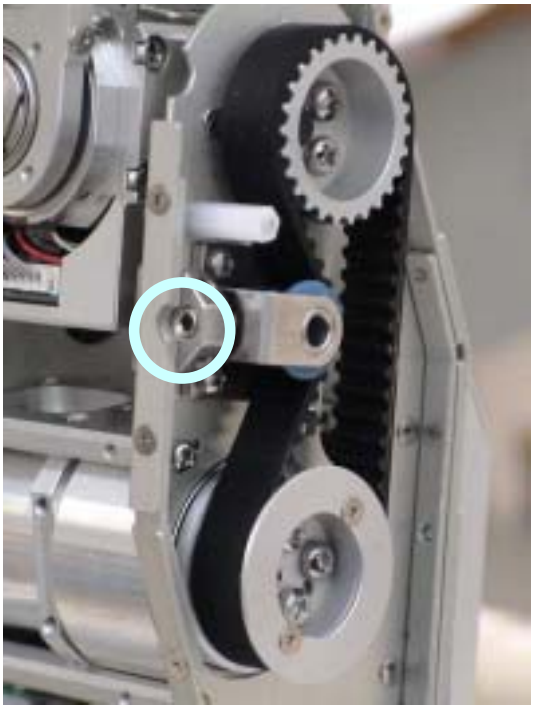


Fig S1.4-5 The usage of a belt adjustment jig

Reference: By our company, the tension of a belt is adjusted by 150-170 [N]. Please adjust according to the conditions of use. (The measuring instrument used: Sound wave formula belt tension meter Gates YUNITTA Asia, Inc.　U-505)

［ｓｔｅｐ５］

Please carry out the increase bundle of the screw which the belt loosened by [step3] after checking that the tension is fully cutting after turning the adjustment screw of a belt adjustment part.

［ｓｔｅｐ６］

The cover removed by [step1] is attached after a belt adjustment end.

S2.1 Command list

S2.1.1 Command for Motor control board and Sensor board

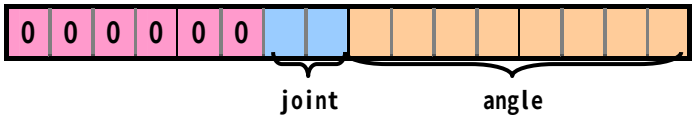
table.s2.1 Command list

	command	parameter	(C)	response		meaning
		byte		byte		
Motor Control board	A	2	Kpp (short×1)	0	-	Position proportional gain
	B	2	Kpv (short×1)	0	-	Velocity proportional gain
	D	2	Kpi (short×1)	0	-	Current proportional gain
	F	2	short×1	0	-	Control mode (0x41:Position, 0x43:Current)
	H	2	short×1	0	-	Velocity limit (0～57)
	I	0	-	0	-	Servo On
	J	0	-	0	-	Servo Off
	L	2	short×1	0	-	Setting the Encorder origin (= 0)
	N	4	short×2	0	-	Encoder Limit (+Limit ,-Limit)
	a	0		6	short×3	Response of the present position & the present velocity
	d	0		6	short×3	Received the setup value of the Position proportional gain & Velocity proportional gain & Current proportional gain.
	e	2	short×1	6	short×3	Target Position value , Response of the present Position
	k	2	short×1	6	short×3	Target Velocity value , Response of the present Position & Velocity
	n	0	-	2	ushort×1	Alarm (1: +limit over , 2 : -limit over)
	o	0	-	0	-	Alarm RESET
	X	0	-	6	ID Version (uchar×6)	Received the Device ID and rhe firmware version.
Sensor board	R	0	-	13	AD0～AD5 (short×6) +IO(uchar×1)	Received the AD conversion data and the IO port status.
	S	1	IO (uchar×1)	0	-	Set the IO port.
	X	0	-	6	ID Version (uchar×6)	Received the Device ID and rhe firmware version.

S2.1.2 Command for RC motor control board(M22～M23)

①RC motor control part

(1) data format



M22
- 00：Head tilt
- 01：Head pan
- 10：Head roll

M23
- 00：Right-hand rotate
- 01：Left-hand rotate
- 10：Right-hand open/close
- 11：Left-hand open/close

	Transmitting data (※) Binary	Decimal	Drive part	Angle angle	direction	note
M22	0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1	15 (= 0+ 15)	Head tilt	-45 °	down	min
	0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0	60 (= 0+ 60)		0 °	front	org
	0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0	120 (= 0+ 75)		15 °	up	max
	0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0	256 (= 256+ 0)	Head pan	-60 °	right	min
	0 0 0 0 0 0 0 1 0 0 1 1 1 1 0 0	316 (= 256+ 60)		0 °	front	org
	0 0 0 0 0 0 0 1 0 1 1 1 1 0 0 0	376 (= 256+ 120)		60 °	left	max
	0 0 0 0 0 0 1 0 0 0 1 1 0 0 1 0	562 (= 512+ 45)	Head roll	-15 °	left	min
	0 0 0 0 0 0 1 0 0 0 1 1 1 1 0 0	572 (= 512+ 60)		0 °	center	org
	0 0 0 0 0 0 1 0 0 1 0 0 0 1 1 0	582 (= 512+ 75)		15 °	right	max
M23	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 (= 0+ 0)	Right hand Rotate (※※)	-60 °		min
	0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0	60 (= 0+ 60)		0 °	center	org
	0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0	120 (= 0+ 120)		60 °		max
	0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0	256 (= 256+ 0)	Left hand Rotate (※※)	-60 °		min
	0 0 0 0 0 0 0 1 0 0 1 1 1 1 0 0	316 (= 256+ 60)		0 °	center	org
	0 0 0 0 0 0 0 1 0 1 1 1 1 0 0 0	376 (= 256+ 120)		60 °		max
	0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0	512 (= 512+ 0)	Right hand Open/close	-60 °	open	min
	0 0 0 0 0 0 1 0 0 0 1 1 1 1 0 0	572 (= 512+ 60)		0 °	center	org
	0 0 0 0 0 0 1 0 0 1 1 1 1 0 0 0	632 (= 512+ 120)		60 °	close	max
	0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0	768 (= 768+ 0)	Left hand Open/close	-60 °	close	min
	0 0 0 0 0 0 1 1 0 0 1 1 1 1 0 0	828 (= 768+ 60)		0 °	center	org
	0 0 0 0 0 0 1 1 0 1 1 1 1 0 0 0	888 (= 768+ 120)		60 °	open	max

※※The actual angle of right-and-left hand revolution increases 1.5 times.

This data is written in.

When sending by the command sendone

1, 3, 22, e, **

(2) The example of a transmitting command

a) A head is turned to the bottom of 45 degrees. : 1,3,22,e,15

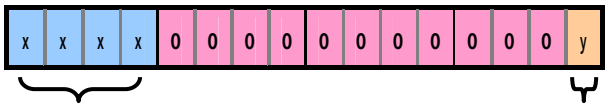
b) A right hand is closed. : 1,3,22,e,632

（3）Notes

In order to transmit specification and the angle of a drive part by one command, it is attached at once and only one motor becomes command transmission. A command interval is set to 10 or more mSecs.

②LED drive part

1）data format



	Transfer data																	drive part	LED ON/OFF
	Binary																Decimal		
M22	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	4096	Right eye	OFF
	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	4097	upper	ON
	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	8192	Right eye	OFF
	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	8193	Low	ON
	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	12288	Right eye	OFF
	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	12289	right	ON
	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16384	Right eye	OFF
	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	16385	Left	ON
	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	32769	Right eye	OFF
	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	32768	rotate	ON
M23	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	4096	Left eye	OFF
	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	4097	upper	ON
	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	8192	Left eye	OFF
	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	8193	Low	ON
	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	12288	Left eye	OFF
	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	12289	Right	ON
	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16384	Left eye	OFF
	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	16385	Left	ON
	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	32768	Left eye	OFF
	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	32769	Rotate	ON

This data is written in.

When sending by the command sendone.

1, 3, 22, e, **

(2) The example of a transmitting command

a) When making LED of a right eye top turn on

1,3,22,e,4097

b) When making LED on the left-hand side of a left eye turn on

1,3,23,e,16385

(3) Notes

- **In order to transmit specification of a drive part, and specification of an angle or LED operation by one command, it is attached at once and only one device becomes command transmission. Please set these command intervals to 10 or more mSecs. A command may be canceled. Rotation of a right eye and a left eye puts out the light automatically, after for [LED] 30 seconds rotates.**

S.2.2 Setting file sample

Regarding contents of initial setting file

Below is sample of sequence mode. Before starting a direct acting servo, such an initialization file is surely transmitted, and initialize the inside of firming.

In case of general command,

Form：[communication interval(ms)],[Header=3],[device No.],[command],[number1],([number 2])

In case of fixed command,

Form：[communication interval],[Header=2],[motor 1command],[motor 2command],...,[sensor 1 command=R],R,R,R

As for the motor command value, meaning varies in the control mode. An command is set up automatically by the control mode.

Send command value corresponding to the control mode. For example, when a motor 2 is set up in the speed control mode, a command to the motor 2 here becomes a target for a speed.

Explanation of HomeReset.csv

2,3,1,A,12	<- Recommendation position proportion gain
2,3,1,B,350	<- Recommendation velocity proportion gain
2,3,1,C,0	<- Fixed setup value (Be sure to setup this value)
2,3,1,D,1	<- Fixed setup value (Be sure to setup this value)
2,3,1,E,0	<- Fixed setup value (Be sure to setup this value)
2,3,1,H,57	<- Velocity limit(Be sure to setup below thin value)
2,3,1,N,6479,-19019	<-Set it up within that value after you confirm allowable movement range of each shaft
2,3,1,F,65	<-The control mode is specified.(position controls is As example)
2,3,1,L,0	<- Encoder zero setting value
2,3,1,e,0	<-Fitted a target value to the encoder zero point.
2,3,2,A,12	<- Set as well as the rest to the motor number 2.
2,3,2,B,350	
：	

Explanation for HomeToWalkInit.csv

10,2,0,0,9405,.....,R,R,R	<- Request sensor data to feed command value to each joint in the interval 10mS.
10,2,0,0,9400,.....,R,R,R	
10,2,0,0,9396,.....,R,R,R	
：	

S2.3 About the renewal of the built-in firmware

A robot is supporting built-in firmware via USB, but if customer by themselves renewal program imprudently, that makes a cause of communication impossibility and re-program becoming impossible, in the worst case, Robot will stop operation. Please contact FJA when renewal firmware.

Also customer is requested to provide below tools when renewal firmware.

- Compiler for the firmware development
  It is necessary when firmware is cording/ compiled. It responds separately, and an incircuit emulator (ICE) may be necessary.
  Supplier：Renesas Technology Corp Model：PS008CAS4-MWR

- HOAP Down load
  It is necessary when down loading firmware via USB. Supply it for Windows98 from Fujitsu.
  Contact FJA for details

S3 Relation of PC for instruction to operate

S3.1 Installation of RTLinux

The procedure of Installation of RTLinux to PC with HOAP is explained.
And, as for Installation to PC except for HOAP, it would be users' (customers) responsibility. In the case, the questions about Installation would not be able to be asked.
The operation except for Attached PC is not confirmed in our company.

S3.1.1 Necessary items for Installation

(1) Attached personal computer
cpu, mouse, keyboard, display, cdrom

(2) LinuxOS
Package **FedoraCore1**(It would be ready by users.)

(3) Linux Kernel source file archive
linux2.4.22.tar.gz (It would be ready by users.)

(4) Kernel configuration file for HOAP
fja.config(It is included in CD with HOAP.)

(5) RTLinux source file archive
rtlinux-3.2-pre3.tar.bz2(It would be ready by users.)

(6) LAN driver of E1000 series
e1000-x.x.x.tar.gz (It would be ready by users.)
Caution: Any visitors other than FMV-630 are unnecessary.

(7) HOAP program for kernel2.4.22
hoap3.tar.gz (It is in the CD with HOAP.)

Note:It is possible to download the file necessary for setup from the site as follow.
Package **FedoraCore1** **http://www.fedoracore.org/**
Linux kernel source files **http://www.kernel.org/**
RTLinux source files **http://www.rtlinux-gpl.org/**
E1000 source files **http://support.intel.com/support/**

S3.1.2 The flow of operation

The installation of RTLinux and operation of confirmation are performed by procedure as follow.

(1) Linux installation
(2) Starting of Linux and GNOME terminal
(3) Preparation of file
(4) Setup of BIOS
(5) RT patch
(6) Compile of Linux Kernel
(7) Compile of RTLinux Modules
(8) Installation of LAN Driver（for FMV-610 ,620）
(9) Installation of PCMCIA Card Service（for PCMCIA Card Adapter）
(10) Installation of HOAP programs

S3.1.3 Operation to install

(1) Linux installation

FedoraCore1 is installed to PC with HOAP.

a) Preparation to install

Please obtain「Package **FedoraCore1**」from the download sites, etc. and enter to the media of CD-R and etc., then create installation CD.
Note:Plural CD-Rs might be installation CDs. (About 3 CDs))

b) Start from CDROM

⇒Please push [F12] from starting monitor after turning on the switch of PC.(In case of FMV)
⇒Please insert the 1st CD created to install to CDROM drive
⇒Please select the「CD-ROM」from keyboard
⇒Please push [Enter]KEY.
Caution:The method(How to change the starting drive) is different. It depends on PCs.

c) The start of Installation

⇒Please push [Enter]KEY when boot: prompt is showed
⇒Please install by the instruction of the monitor.

Please check the site as following about the method(How to install).
http://www.redhat.com/

Note:It is installed and the operation is confirmed in the conditions as following in our company.

Language Selection japanese
Keyboard Selection jp106
Mouse Selection Generic-2 Button Mouse(PS/2)
Installation Type ***Custom***
Disk Partitioninig Setup Disk Druid
Mount Point: / File system type: ext2 Size: over 2G
Mount Point: File system type: swap Size: over 256
Boot Loader Configuration
Use GRUB
MBR
LBA32N
Firewall Configuration No firewall
Language Support Japanese
Time Zone Selection GMT
Package Group Selection
Sound and Multimedia Support
Network Support
Software Development
Kernel Development

(2) Starting of Linux and GNOME terminal

Installed Linux from (1) is opened, the file necessary for operation is copied.

a) Starting of Linux

PC is opened, Please select Linux by [cross Key] on the GRUB loader monitor in the time to start.

b) LOGIN

Login is carried out by root.

local host login : **root** ↓
password: ******** ↓
↑ Password setup on the time to install Linux

Note:When the login is performed in the user's name except for root and after started the GNOME terminal, it is possible to change the right to "root" by "su command".

c) Starting of GNOME

After the login, the command as follow is input, and GNOME is started.

startx ↓

Note:Though the warning message for login by "root" is showed, please push[Enter]Key and continue.

d) Starting of GNOME terminal

GNOME terminal is started after the starting GNOME window.

⇒Please click the GNOME terminal icon by.

Subsequent work is done in form of inputting a command into a GNOME terminal from a keyboard.

The command for input to the terminal is written in bigger (wider) size character.

And the last "↓" on the command line to input means "[Enter] Key.

(3) Preparation of the files

The files necessary for the installation are prepared.

a) Download of files

Please obtain the file as follow from the download site, and enter the file to the CD-R.

- linux-2.4.22.tar.gz
- rtlinux-3.2-pre3.tar.bz2

b) Mount of CDROM

Prepared CD is mounted to CDROM drive.

⇒Please insert CDROM to the drive.

mount /dev/cdrom /mnt/cdrom ↓

c) Copy of source file

Done download file is copied to "/usr/src".

cp /mnt/cdrom/linux-2.4.18.tar.gz /usr/src/. ↓

cp /mnt/cdrom/rtlinux-3.2-pre1.tar.bz2 /usr/src/. ↓

d) Unmount of CDROM

Mount of CDROM is taken off and it is taken out from the drive.

umount /mnt/cdrom ↓

⇒Please eject CDROM from the drive.

e) Unpack the source files.

The copied files are decompressed.

cd /usr/src ↓

tar zxvf linux-2.4.22.tar.gz ↓

bunzip2 -d rtlinux-3.2-pre3.tar.bz2 ↓

tar xvf rtlinux-3.2-pre3.tar ↓

Note:When an error comes out at the time of decompression of a file, please redo from download.

f) Movement and the link of a kernel source file

Kernel sauce is moved to RT sauce and a link is created.

mv /usr/src/linux /usr/src/rtlinux-3.2-pre3/. ↓

ln –s /usr/src/rtlinux-3.2-pre3 /usr/src/rtlinux ↓

ln –s /usr/src/rtlinux/linux /usr/src/linux ↓

g) Preparation of configuration file

Configuration file is copied from CD with HOAP to Kernel source.

⇒Please put CD of attachment in HOAP into a CDROM drive.

mount /dev/cdrom /mnt/cdrom ↓

cp /mnt/cdrom/fja.config /usr/src/rtlinux-3.2pre3/linux/.config ↓

umount /mnt/cdrom ↓

⇒Please eject CDROM from a drive.

(4) Setup BIOS

a) Display of BIOS monitor

When FMV is started, push [F2] key, and BIOS monitor is displayed.

b) The contents of a setting of BIOS

[Adbanced]
[Serial/Parallel Port configurations]↓
Serial Port1:[Disabled]
Serial Port2:[Disabled]
Parallel Port:[Disabled]
[Internal Device Configurations]↓
USB Controller:[Both]
USB2.0 Controller:[Disabled]
USB Legacy Emulation[Disabled]

c) Saving and Exit

[Exit]
[Exit Saving Changes]↓

(5) RT patch

Patch the kernel

a) Start of Linux

⇒Please select **FedoraCore(2.4.22)** from GRUB.

b) Unpack the files

cd /usr/src/rtlinux-3.2-pre3/patches ↓

bunzip2 -d kernel_patch-2.4.22-rtl3.2-pre3.bz2 ↓

c) Patch

cd /usr/src/rtlinux-3.2-pre3/linux ↓

patch –p1 < ../patches/kernel_patch-2.4.22-rtl3.2-pre3 ↓

(6) Compile the RTLinux kernel

Compile and install the kernel and modules.

a) Start of Linux

⇒Please select **FedoraCore (2.4.22)** from GRUB.

Note:If it has been already starting on RT Linux(2.4.22-rtl3.2_pre3), it is unnecessary to re-start it.

b) Configure the kernel

cd /usr/src/rtlinux-3.2-pre3/linux ↓

make xconfig ↓

⇒It is setup as follow from configuration menu.

①[Processor type and features] (Refer to "Caution")
Set " Symmetric multi-processing support " to [n]
②[File systems]
Set " Ext3 journalling file system support " to [y]
③[Code maturity level options]
Set " Prompt for development and/or incomplete code/drivers " to [y]

⇒ Please select [Save and Exit].

⇒ Please select [OK].

Caution:①Please setup it in the case of FMV-C630.

c) Compile and install

make clean ↓

make dep ↓

make bzImage ↓

make modules ↓

make modules_install ↓

make install ↓

d) Default Setup of BOOT OS

GRUB configuration file is edited and RTLinux is setup to OS of default.

gedit /boot/grub/grub.conf ↓

⇒The line of「default」is changed.

default=0

⇒Plesse select the [save] Button.

⇒Plesse select the [file(F)]⇒[Exit(E)]

Note: In the time after the second compile, three lines are deleted as follow.

Title Red Hat Linux(2.4.18-rtl3.2-pre1)
root (hd0,1)
kernel /vmlinuz-2.4.18-rtl3.2-pre1 ro root=/dev/hda3 hdc=ide-scsi

e) reboot

It reboots from a new kernel.

reboot ↓

(7) Compile and install the RTLinux modules.

a) reboots from a RTLinux kernel

⇒Please select the RT Linux(2.4.22-rtl3.2_pre3).

b) Configure

cd /usr/src/rtlinux-3.2-pre3 ↓

make xconfig ↓

⇒Do you agree to the terms of this license[y/n]?

y ↓

⇒Please select [Save and Exit].

⇒Please select [yes].

c) Compile and install

make clean ↓

make dep ↓

make all ↓

make dirs_install ↓

make install ↓

d) The check of RTLinux of operation

rtlinux start ↓

⇒Please check the following displays.

Scheme(-)not loaded, (+) loaded

(+)mbuff

(+)rtl

(+)rtl_fifo

(+)rtl_posixio

(+)rtl_sched

(+)rtl_time

(-)rtl_sched

(-)rtl_time

e) The stop check of RTLinux

rtlinux stop ↓

⇒Please check the following displays.

Scheme(-)not loaded, (+) loaded

(-)mbuff

(-)rtl

(-)rtl_fifo

(-)rtl_posixio

(8) Installation of a LAN driver（FMV-630 only）

a) reboots from a RTLinux kernel

⇒Please select RT Linux(2.4.22-rtl3.2_pre3)

b) The copy of E1000 driver

⇒Please insert CD in a CDROM drive.

mount /dev/cdrom /mnt/cdrom ↓

cp /mnt/cdrom/e1000-x.x.x.x.tar.gz /usr/local/. ↓

umount /mnt/cdrom ↓

⇒Please eject CDROM from a drive.

c) Unpack the source files.

cd /usr/local ↓

tar zxvf e1000-x.x.x.x.tar.gz ↓

d) Installation

cd /usr/local/e1000-x.x.x.x/src↓

make install↓

e) Setup

⇒[Program]⇒[System]⇒[Network setup]

⇒Start

⇒Close

(9) Installation of HOAP program

a) Copy of HOAP program

⇒Please insert the HOAP attachment CD in a CDROM drive.

mount /dev/cdrom /mnt/cdrom ↓

cp /mnt/cdrom/hoap3.tar.gz /usr/local/. ↓

umount /mnt/cdrom ↓

⇒Please eject CDROM from a drive.

b) Unpack the HOAP program

cd /usr/local ↓

tar zxvf hoap3.tar.gz ↓

c) The check of HOAP program

cd /usr/local/hoap3/modules ↓

rtlinux start rt_ctlmodule.o ↓

⇒Please check the following displays.

(+) rt_ctlmodule

/usr/local/hoap3/bin/rt_ctlapp ↓

⇒Please check the following displays.

0,0,00:$

$q

S3.2 Wireless mode process

Preparation

1. Read the manual of the wireless LAN access point, and SSID of the wireless LAN access point is set up in "GeoWave".
2. The Compact Flash which puts LinuxOS is inserted into the slot of the robot.
   Insert Wireless LAN card into Robot.
3. Switch ON robot power source.
   Provide the PC for remote access which connected a wireless LAN access point and a network.
   It cares about either Windows or Linux.
   Refer to [S3.2.1 wireless motion process note] regarding motion process of Linux.

※Software loading CompactFlash is setup to communicate with the access point of SSID is "GeoWave" and IP address is 10.25.184.151 by the default.
Network setting can be re-write in case of necessity. Refer to [S3.3 : Compact Flash setup] .

Wireless motion

1. From remote access PC, remote login is carried out to the robot loading PC.
   A command prompt is started.
   telnet 10.59.145.197 is inputted and a return is carried out.
2. Login as User **guest**、Password **guest**. After Login、
   Input as **$ su**
   Then return, input password **default**, log in with root authority.
   The rest can be operated with a same command that is operating method from local user PC.
3. after finishing operation, input as
   #poff
   LED blink of both eyes rotates. When rotation stops and LED puts out the light, it is the completion of a shutdown.
   It has the possibility that the contents of Compact Flash crash when a robot power supply is turned off without doing this. In case compact flash crashed and do not start up, re-install process is referred to [S3.3 compact flash set up].

S3.2.1 Wireless mode process

1.Turn to the " X" of 「MDI/MDI-X」switch of the access point. And connect between the remote access PC and the access point by LAN cable. Turn on the PC power.

2.When Fodora Core is started on PC, login like the wired mode.

Local host login : **root**

Password : **default**

3.Input # **startx**

4. Please check that surrounding LED of a robot's eyes is on.

There is this several minutes case after turning on a robot until LED of an eye lights up.

5.Start terminal emulator for GNOME

6.Input the following in this window,

```
# telnet 10.59.145.197
id       : guest
password : guest
$ su
password : default
```

7.Input the following in this window,

```
# .. /bin/rt_ctlapp2
0.0.00:$    (<- response prompt)
```

Here, pressing a return key is continued.

```
5.27.xx:$
```

Please check becoming. (The value of xx is unfixed.)

8. Please newly start one more terminal screen. (Let the screen which was opening this screen from a terminal 2 and before be a terminal 1.)

9. It logs in in the procedure of 6. like a terminal 1 in a terminal 2.

10. A robot is hung, it hangs to a lifting jigu, and hand and foot are extended lightly.
(Please refer to Fig.S3.2-1)

11. The following commands are executed in a terminal 2.

```
# cd /usr/local/hoap3/data
# sh setup.sh
    (Initial setting is performed.)
# sh_walk.sh
    (Sample walk operation is performed.)
```

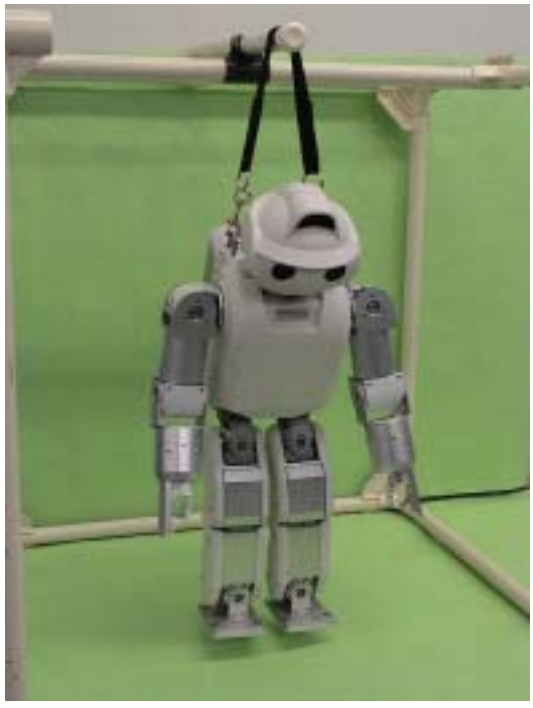


Fig.S3.2-1 State which hung the robot

How to shutdown

1.Hung the robot to lifting jig.

2.Input following by Window 2

 # sh svoff.sh

3.Finish the monitor program by Window1.

5.27.xx**:$ q (Return)**

4.shutdown by Window1.

 # poff

5.Close both windows.

 # exit

6. After rotation of LED of an eye disappears, please drop a robot's power supply. If it drops early, the program in CF card will crash and carry out.

7. Remote access PC is carried out Power Down.

S3.3 Compact Flash set up

S.3.3.1. Necessary parts

(1) PC which is attached PC card slot

(2) Compact Flash (1GByte)

(3) CD-ROM of HOAP

S.3.3.2. Process outline

(1) Provide PC to write in CF.

(2) Cut partition of CF

(3) Copy OS in CF

(4) Make possible CF booted .

(5) Copy the HOAP software in CF.

(6) Boot from CF, and check if it is possible to login via network.

S.3.3.3 Provide PC to write in CF.

Provide PC which is attached PC card slot.

Install **FedoraCore1** on PC. (refer to S.3.1)

S.3.3.4 Cut partition of CF

Login PC that finished set up at “S3.3.2” by root authority.

Click on the icon of "display and footprint" type at lower of side of screen , and a CF is set on the CF adapter, and that is inserted into the PC card slot, terminal emulator for GNOME is started.

At terminal emulator, input as

tail /var/log/messages

It is checked the CF which IDE device was recognized as.

kernel: idc_cs: hdc: Vcc=3.3, Vpp = 0.0

cardmgr[*]: executing: ‘./ide start hdc’

If write above, it is recognized as hdc.

less /proc/partition

Like to see above, find out which IDE device is recognized.

Input as **/usr/sbin/cdfidk /dev/hdc**

Then Start cdfdisk.

Chosen deletion with an arrow (→) key and if it has an existent territory, chosen with a ↓ key and deleted.

Chosen “A new preparation” → “basic area”.

Size : (Return)

「From beginning」(Return)

「Boot possible」(Return)

It carries out and all capacity is created as a basic partition.

It checks that it is as follows

hdc1 basic boot area Linux 1000.00

(It is a near value although not exactly set to 1G.)

and "writing" is chosen by the -> key.

Is it all right although partition information is written in a disk?:yes(Return)

It comes out and writing is performed.

The domain table was written in the disk.

If it comes out, an end will be chosen by the -> key and cfdisk will be ended.

CompactFlash is once extracted and is inserted again.

/sbin/mke2fs /dev/hdc1

It inputs and ext2 file system is created.

/sbin/mke2fs /dev/hdc2

It inputs and a swap space is created.

S3.3.5 Copy OS on CF

mkdir /mnt/flash

mount /dev/hdc1 /mnt/flash

Input as above, and mount compact flash on /mnt/flash

Insert CD of HOAP into cd-rom drive.

Double-click on the icon of the CD form at left side of the desk-top computer (screen), and indicate the contents of the CD.

Chosen below from indicated window's menu,

linux-flash.tar.gz

then, click right side. Chosen copy, purposed place, and write

/tmp

Push OK button.

cd /tmp

tar zxv -C /mnt/flash -f linux_flash.tar.gz

Input above and extract OS to CF.

S3.3.6 Make CF to boot possible

cd /mnt/flash; /usr/sbin/chroot . sbin/lilo –C etc/lilo.conf.flash

Input as above, write in master boot record of CF make boot possible.

Added linux-2.4.22-rtl *

Succeed if above is indicated.

S3.3.7 Copy HOAP software to CF

Insert HOAP CD into cd-rom drive.

Double-click on the icon of the CD form at left side of the desk-top computer (screen), and indicate the contents of the CD.

Chosen below from indicated window's menu.

hoap_ohci_release.tar.gz

data.tar.gz

then, click right side. Chosen copy, purposed place, and write

/tmp

Push OK button.

tar zxv -C /mnt/win/usr/local/ -f hoap_ohci_release.tar.gz

input above, install HOAP software into /mnt/win/usr/local

Appears hoap.ohci directory in /usr/local/, Symbol click "ln -s hoap.ohci hoap"

Also、input as below, and install walking data.

tar zxv -C /mnt/win/usr/local/hoap -f data.tar.gz

Input as below and unmount CF.

umount /dev/hdc1

S3.3.8 Start from CF, and check whether it can be log in via network.

Set up of the CF is:

ip address：**10.59.145.197**

netmask：255.255.255.0

gateway：10.59.145.1

hostname：hoap3

Above setup is written in "/etc/sysconfig/network" file , "/etc/sysconfig/network-scripts/ifcfg-eht0" file, and "/etc/hosts" file.

Change them if necessary.

Please connect with an Ethernet hub PC for a remote access (Windows or Linux is also good) which the Ethernet card attached, and also connect a wireless LAN access point with the hub. From remote access PC, remote login is carried out to the robot loading PC.

Please start a command prompt.

telnet 10.59.145.197

Please input above and press a return key.

It can log in with User guest and Password guest. If login is possible,

su

Please press an input and a return key, enter a password as default and log in by root authority. The rest can be similarly operated by the same command as the operation method from a local user personal computer.

Please input exit, when you end remote login. before stopping a robot's power supply - surely

 # /sbin/shutdown -h now

S3.4 Explanation for sequence data file.

Following data file of sequence motion is including in “/usr/local/hoap3/data” directory.

It can be used as below by sendseq program.

/usr/local/hoap3/bin/sendseq < *.csv

File name	Explanation
AllServoOff.csv	All joints Servo off
AllServoOn.csv	All joints Servo On
HomeReset3o.csv	It transmits in the state where it installed in the initial posture setting jig, and is a KAUNTAPU reset + SABO gain + movable range setup.
m03.csv	ZMP walking

S4 Caution item for RT-Linux

When power supply is turned off compulsively due to the blackout or the hang-up during the use, It has the possibility to damage the file-system of Linux.

When it was started and it stopped by the following message.

(as example here, in case of hda2 of IDE)

```
Give root password for maintenance
 (or type Control-D for normal startup):
```

The system this message judged that there was a wrong point in the ext3 file system, they demanded to have a disk check by the manual.

Log in by root password (initial setting is by default) to type 「Ctrl」＋「D」

If following is /dev/hda**2**(IDE hard disc of first unit), check disc to input following.

If there is a question a middle of it (like send fix?), reply (Y) to all.

```
# fsck -t ext2 /dev/hda2          (partition of /boot)

# fsck -t ext2 /dev/hda5          (partition of /)
```

S5 Current control

S5.1 How to use of the current control mode

Use the command “D” for setting.

Argument is 16bit data, and following the bit-asign.

At bit15～bit12, specificate to this command. (bit15,bit14 are Don't care)

bit15	bit14	bit13	bit12	
x	x	0	0	setting of the current proportional gain
x	x	0	1	setting of the CCW origin
x	x	1	0	setting of the CW origin
x	x	1	1	setting of the same origin both CW and CCW

At bit11～bit1, set 0～4096

bit11	bit10	bit1	bit0	
0	0	0	0	0
0	0	0	1	1
0	0	1	1	2
	..			
1	1	1	1	4096

For example, When argument is 0x1015, CCW origin is 21.

The origin is set the current before begining to move the motor.

These values need to set up a proper setting value with the system according to the place and use situation of a motor.

S5.2 Sample program of thecurrent control

This sample is

For M01 motor

Target current value is 30. (About 1Ampere at Type-3 motor)

When motor moves to limit, it is servo off.

If rotation is blocked, the target current continues flowing.

```
2 3 1 D    24          ← setting of the current proportional gain (18H)
2 3 1 D  4117          ← setting of the CCW origin (1015H)
2 3 1 D  8212          ← setting of the CW origin (2014H)
2 3 1 N 10000 -10000   ← setting of the encorder limitter
2 3 1 F    67          ← setting of the current control mode (43H)
2 3 1 k    30          ← setting of the target current value
2 3 1 L     0          ← setting of the encorder origin
2 3 1 I                ← Servo On
```

S5.3 Sample of the calculation

The following is Sample that the calculation of the current.

the current proportional gain is "18H",

the CCW origin is "15H",

the CW origin is "14H"

TYPE－2 motor

Target current value： I＝（1．25／70）×k ＋ 0．29

TYPE－3 motor

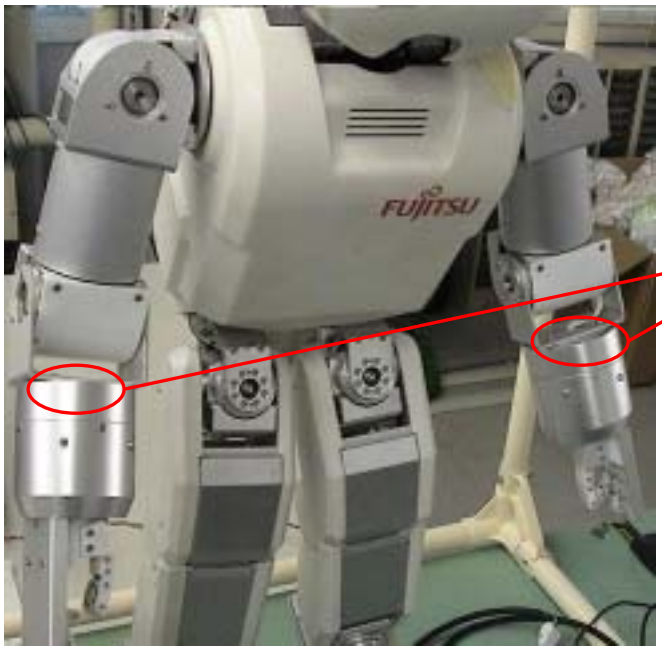
Target current value： I＝（1．55／70）×k ＋ 0．29

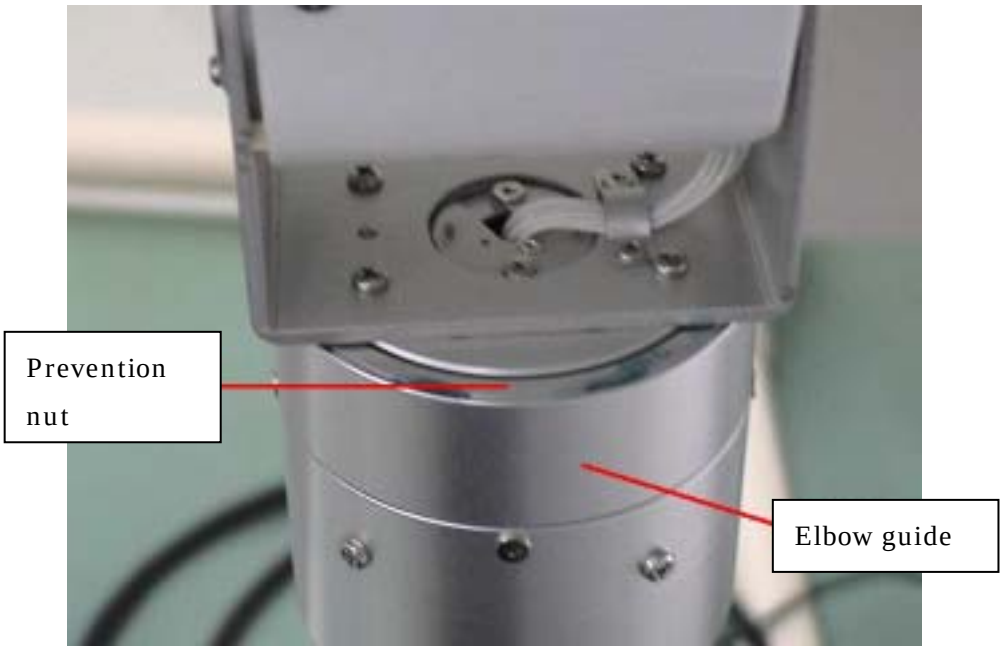
※ It is necessary to change the above-mentioned constant according to the individual difference of a motor.

In addition, Set the target current value below to 3Ampere.

S 6　The attachment check of an arm portion

In the state at the time of shipment, although it has taken all possible measures, if you use a fall etc., repeating, conclusion of the next part may loosen. Please confirm before the start of operation on the 1st etc.





Check part enlargement

An elbow guide is seen from the front, and when a prevention nut seems to have come out upwards, conclusion of a prevention nut and an elbow guide is loosening. To the position where a prevention nut disappears, please fasten an elbow guide and be crowded.