Omron Adept Mobile Robots

Development Platforms for Research and Education

Pioneer 3

Operations Manual

P3DX-SH , P3AT-SH with ARCOS

P3X0001, P3T0001

Revision 6.5 ● April 7, 2017

About Omron Adept MobileRobots

In 1995, ActivMedia, Inc. ventured with Real World Interface, Inc. to launch the Pioneer Mobile Robot. In 1999, ActivMedia Robotics began evolving the Pioneer line of mobile robot platforms for research, development, commercial and industrial use. In 2006, ActivMedia Robotics, LLC, became MobileRobots Inc. In 2010, MobileRobots Inc. was acquired and merged with Omron Adept Technology, Inc. In 2015, Adept was acquired by Omron, a global electronics and industrial automation company.

As part of Omron Adept Technologies and Omron, we continue to develop and support the Pioneer robots for research, education and custom R&D. Even though the names have changed, our devotion to provide customizable, high quality, durable platforms for custom robotics development applications has not.

Pioneer 1 (1995)

Omron Adept MobileRobots

10 Columbia Dr.
Amherst, NH, 03031
USA

1 (603) 881-7960
http://www.mobilerobots.com

Important Safety Instructions

- ✓ Read the installation and operations instructions before using the equipment.
- ✓ Avoid using power extension cords.
- ✓ To prevent fire or shock hazard, do not expose the equipment to rain or moisture.
- ✓ Refrain from opening the unit or any of its accessories.
- ✓ Keep wheels away from long hair or fur.
- ✓ Never access the interior of the robot with charger attached or batteries inserted.

Inappropriate Operation

Inappropriate operation voids your warranty! Inappropriate operation includes, but is not limited to:

- ✓ Dropping the robot, running it off a ledge, or otherwise operating it in an irresponsible manner
- ✓ Overloading the robot above its payload capacity
- ✓ Getting the robot wet
- ✓ Continuing to run the robot after hair, yarn, string, or any other items have become wound around the robot's axles or wheels
- ✓ Opening the robot with charger attached and/or batteries inserted
- ✓ All other forms of inappropriate operation or care

**Use authorized parts *ONLY*;
warranty void otherwise.**

Contents

Chapter 1 Introduction

Congratulations on your purchase and welcome to the rapidly growing community of developers and enthusiasts of Pioneer intelligent mobile platforms.

This manual provides both the general and technical details you need to operate your robot and to begin developing your own robotics software.

Note: For operation of previous versions of robots which use the Siemens C166- or Hitachi H8S-based robot microcontrollers, original motor-power boards and support systems, please contact sales@MobileRobots.com or access our support website: `http://robots.MobileRobots.com` for their related documentation.

ROBOT PACKAGES

Our experienced manufacturing staff put your mobile robot and accessories through a "burn in" period and carefully tested them before shipping the products to you. In addition to the companion resources listed above, we warrant your Pioneer 3 platform and our manufactured accessories against mechanical, electronic, and labor defects for one year. Third-party accessories are warranted by their manufacturers, typically for 90 days.

Even though we've made every effort to make your robot package complete, please check the components carefully after you unpack them from the shipping crate.

Included Components (all robots)

- ✓ One fully assembled mobile robot, battery or batteries, and charger
  - Differential drive system
  - Wheel encoders
  - Dedicated robot motion controller (microcontroller)
  - Motor/Power distribution board
  - Power and battery level indicators
  - AUX Power switches on side control panel
  - RESET and MOTORS pushbutton controls on side panel
  - Piezo buzzer
  - Optional onboard computer IO connections and power button
- ✓ CD, DVD or USB device containing licensed copies of Pioneer SDK software and documentation
- ✓ Any necessary wrenches, mounting screws, fuses and other replacement parts
- ✓ Manual(s)
- ✓ Supplementary documentation for accessories and options
- ✓ Registration and Account Sheet

Optional Components, Parts and Attachments (partial list)

Contact sales@mobilerobots.com or visit http://www.mobilerobots.com for more information on ordering replacement or additional optional accessory parts. Refer to additional documentation included with your shipment and available at http://robots.mobilerobots.com for more information on accessories and options ordered with your robot.

Some options are only available on certain robot platforms.

- ✓ Onboard internal onboard single-board computer
- ✓ Additional batteries
- ✓ Optional sonar

- ✓ Laser range finder (with navigation software development kits for autonomous navigation of the robot)
- ✓ Gripper
- ✓ Manipulator Arm with gripper
- ✓ Stereo Vision Systems
- ✓ Pan-Tilt-Zoom Camera
- ✓ Global Positioning System
- ✓ Heading-correction accelerometer gyro (installed inside robot)
- ✓ Front and/or rear bumper switch assemblies, or replacement switches.
- ✓ USB-Serial communications/configuration adapter cable
- ✓ Mounting brackets and plates
- ✓ E-Stop button
- ✓ Replacement or additional top plate(s)
- ✓ Replacement wheels
- ✓ Indoor-use wheels (Pioneer 3-AT)
- ✓ Optional 2-axis, 2-button joystick for direct manual control of robot

IO and Power Connections

Your Pioneer 3 robot has a variety of expansion power and I/O ports for integration of a client PC, sensors, and other accessories.

- ✓ 3 RS-232 auxilliary serial ports (AUX serial) configurable from 9.6 to 115.2 kilobaud, accessible through robot microcontroller
- ✓ 1 “HOST” (or “SERIAL”) client software communications connection (internal and external serial ports)
- ✓ Maximum 4 optional ultrasonic “SONAR” sensor arrays of 8 ultrasonic transducers each
- ✓ inputs for optional bumper switches
- ✓ 16 digital I/O lines for accessories or customer use
- ✓ Analog input
- ✓ 5 and 12 V DC power
- ✓ Optional heading correction gyro
- ✓ Optional 2-axis, 2-button joystick for direct manual control of robot

SOFTWARE AND PIONEER SDK

The Pioneer SDK is a collection of libraries and applications that come with every Pioneer mobile robot and with selected accessories. The standard Pioneer SDK bundled with every robot at no extra charge includes the open-source ARIA and ArNetworking, the MobileEyes and Mapper3-Basic network GUI applications, SONARNL and MobileSim.

ARIA provides an interface and framework for controlling and receiving data from all Pioneers, as well as most accessories (some devices also have separate interface libraries). ARIA also has utilities useful for writing robot control software as well as tools for writing cross-platform (Windows and Linux) code and support for network sockets and threads. ArNetworking provides a simple, extensible framework for client-server network programming.

MobileSim is an open-source application based on Stage that Omron Adept MobileRobots engineers have customized and extended to best simulate all Pioneer platforms and many accessories.

MobileEyes, enabled through ARIA and ArNetworking, is a GUI application for configuration, operation and monitoring of your Pioneer platform over the network.

The **SONARNL** libraries enable your SONAR-based Pioneer platforms to localize and autonomously navigate indoor spaces. Make your working maps for SONARNL with the free GUI Mapper 3-Basic application.

Optional, typically accessory-bundled Pioneer SDK libraries include **ARNL** and **MOGS**. ARNL enables a much more robust, laser-based autonomous localization and navigation. ARNL is the best-in-class software foundation for Omron Adept MobileRobots' Motivity commercial- and industrial-ready mobile localization and navigation systems. Use MOGS to fuse laser and DGPS sensor data to guide your Pioneer robots outdoors.

Several other robotics applications development environments also have emerged to support Pioneer mobile robots, including Ayllu, Player/Stage, Carmen, Microsoft Robotics Developer Studio and Robot Operating System (ROS). See documentation provided with the development environment for more information, including documentation on using Pioneer robots.

ADDITIONAL RESOURCES

New customers get three additional and valuable resources:

- ✓ A private account on our support Internet website for downloading software, updates, and manuals
- ✓ Access to private newsgroups
- ✓ Direct access to the Pioneer technical support team

Support Website

For software updates, manuals, and a knowledge base of information, or to contact technical support staff, visit:

`http://support.mobilerobots.com`

Some areas of the website are restricted to licensed customers. To gain access, enter the username and password written on the *Registration & Account Sheet* that accompanied your robot.

Newsgroups

We maintain several email-based newsgroups through which robot owners share ideas, software, and questions about the robot. Visit the support `http://support.mobilerobots.com` website for more details. To sign up for pioneer-users, for example, send an e-mail message to the `–requests` automated newsgroup server:

To: pioneer-users-requests@MobileRobots.com

From: <your return e-mail address goes here>

Subject: <choose one command:>

help (returns instructions)

lists (returns list of newsgroups)

subscribe

unsubscribe

Our e-mail list server will respond automatically. After you subscribe, e-mail your comments, suggestions, and questions intended for the worldwide community of Pioneer users:[1]

To: pioneer-users@MobileRobots.com

From: <your return e-mail address goes here>

Subject: <something of interest to pioneer users>

[1] Note: Leave out the –requests part of the email address when sending messages to the newsgroup.

Access to the `pioneer-users` e-mail newslist is limited to subscribers, so your address is safe from spam. However, the list currently is unmoderated, so please confine your comments and inquiries to issues concerning the operation and programming of Pioneer platforms.

Support

Have a problem? Can't find the answer in this or any of the accompanying manuals? Or do you know a way that we might improve our robots? Share your thoughts and questions with us from the online form at the support website:

`http://robots.MobileRobots.com/techsupport`

or by email:

`support@MobileRobots.com`

Please include your robot's ***serial number*** (look for it beside the `Main Power` switch)—we often need to understand your robot's configuration to best answer your question.

Tell us your robot's SERIAL NUMBER.

Your message goes directly to the technical support team. There a staff member will help you or point you to a place where you can find help.

Because this is a support option, not a general-interest newsgroup like `pioneer-users`, we reserve the option to reply only to questions about problems with your robot or software.

Use authorized parts *ONLY*;
warranty void otherwise.

Chapter 2 Specifications & Controls

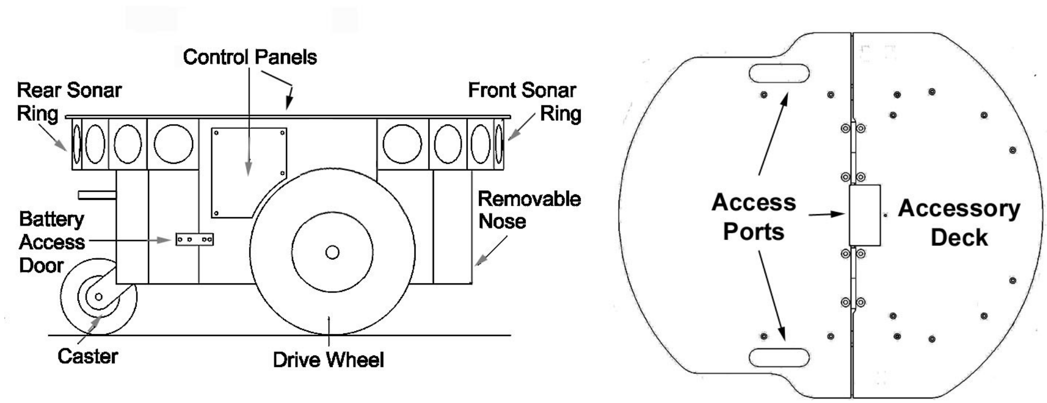


Figure 1. Pioneer 3-DX features

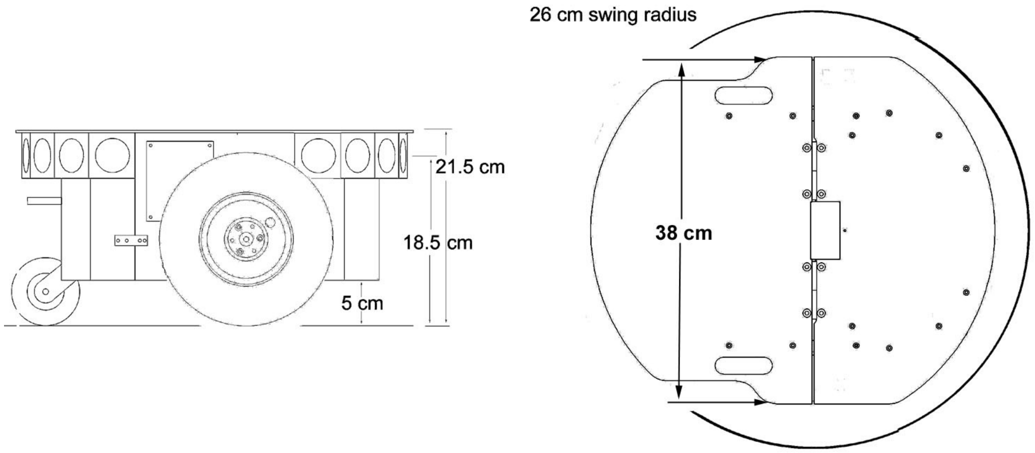


Figure 2. Pioneer 3-DX's physical dimensions and swing radius.

Physical Characteristics and Components

Pioneer robots are composed of several main parts:

- ✓ Deck
- ✓ Motor Stop Button (optional on DX)
- ✓ User Control Panel
- ✓ Body, Nose, and Accessory Panels
- ✓ Optional SONAR Array(s)
- ✓ Motors, Wheels, and Encoders
- ✓ Batteries and Power

DECK

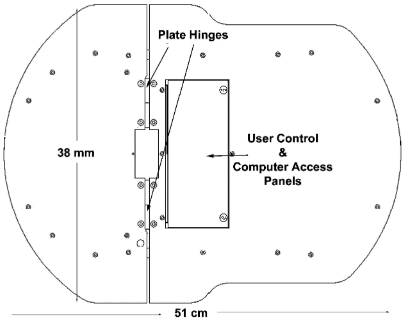


Figure 3. Pioneer 3-AT's console and hinged deck

Pioneer 3 has a hinged top-plate or deck which gives you access to internal components of the robot, and provides a surface for mounting accessory devices and other equipment . Slots on each side of the deck plate let you conveniently route cables to the accessory connectors on the side panels of the robot. A removable plug in the middle of the deck plate allows cables to enter the interior of the robot.

Equipment may be attached to the deck or body using the predrilled mounting holes, or by removing the top plate and cutting, drilling or customizing as needed.

Warning: Do not drill or cut the top plate while mounted on the robot. Always remove the plate from the robot before modifying.

Important: When mounting accessories, you should try to center the robot's payload over the drive wheels. A full complement of batteries helps balance the robot, too.

Warning: always remove the deck (top plate) from the robot before drilling or cutting!

E-STOP BUTTON

A STOP button is included standard with Pioneer 3-AT and optionally with Pioneer 3-DX. Press to immediately disengage the robot's motor power. It will also cause a stall state, indicated in flags available to user software and by beeping sound from the onboard piezo speaker (unless sounds have been disabled).

Press the STOP button again to re-enable motor power.

USER CONTROL PANEL

The User Control Panel is where you have access to controls for the ARCOS-based onboard robot microcontroller. This panel is on the top deck of the Pioneer 3 AT, and is on the left sidepanel of the Pioneer 3 DX.

Power and Status Indicators

The red PWR LED is lit whenever main power is applied to the robot. The green STAT LED state depends on the operating mode and other conditions. It flashes slowly when the microcontroller is awaiting a connection with a client and flashes quickly when in joystick drive mode or when connected with a client and the motors are engaged. It also flashes moderately fast when the microcontroller is in maintenance mode.

The BATTERY LED's apparent color depends on your robot's battery voltage: green when fully charged (>12.5 volts) through orange, and finally red when the voltage drops below 11.5. When in maintenance mode, the BATTERY LED glows bright red only, regardless of battery charge.

Buzzer

A built-in piezo buzzer provides audible clues to the robot's state, such as upon successful startup of the microcontroller and a client connection. The SAY command lets you program the buzzer, too, to play your own MIDI sounds (see Client Commands on page 27 for details).

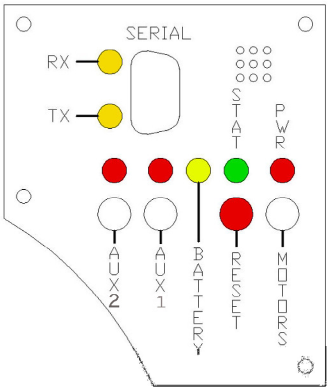


Figure 4. P3-DX User Control Panel

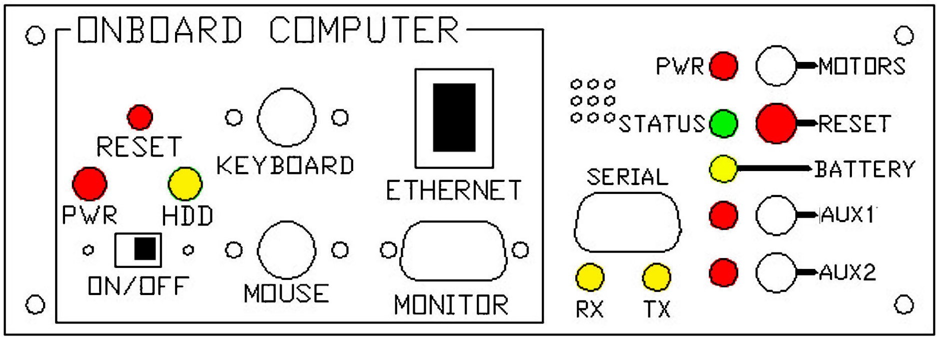


Figure 5. P3-AT computer and user controls

Serial Port

The host SERIAL connector, with incoming and outgoing data indicator LEDs (RX and TX, respectively), is through where you may interact with the ARCOS microcontroller from an offboard computer for tethered client-server control and for microcontroller software maintenance. The port is shared internally by the HOST serial port, to which we connect the onboard computer or an Ethernet-to-serial device. Either the SERIAL or HOST connector may be used for client-server and maintenance mode communication with the microcontroller.

To avoid communication conflicts, digital switching circuitry disables the internal HOST serial port if the attached serial device hasn't opened the port. However, serial port interference will be a problem if the HOST and User Control SERIAL ports are both occupied and engaged. Accordingly, remove the cable from the User Control SERIAL port if you plan to connect with the microcontroller through the HOST port.

In particular, if you have a serial cable connected to the User Control Panel SERIAL port, with the attached PC has that serial port opened for communications, and you then reset or power up the robot and microcontroller, ARCOS automatically goes into maintenance mode.

Power Switches

The AUX1 and AUX2 switches on the User Control Panel are pushbuttons which engage or disengage power to 5 and 12 VDC connectors on the Motor-Power board to which we or you attach power for various accessories. For example, 12 VDC power for the PTZ camera typically gets switched via the AUX1 pushbutton. See *Appendix B* for power connections. Respective red LEDs indicate when power is ON.

Reset and Motors

The red RESET pushbutton acts to unconditionally reset the microcontroller, disabling any active connections or attached devices, including the motors.

The white MOTORS pushbutton's actions depend on the state of the microcontroller. When connected with a client, push it to enable and disable the motors manually, as its label implies.[2]

To manually engage ARCOS maintenance mode, press and hold the white MOTORS button, press and release the red RESET button, then release MOTORS. Note that while this manual operation was required to engage maintenance mode with previous robot microcontrollers, it is no longer necessary with ARCOS.

Body, Nose and Accessory Panels

Your Pioneer 3's sturdy, but lightweight aluminum body houses the batteries, drive motors, electronics and other common components, including the front and rear SONAR arrays. The body also has sufficient room, with power and signal connectors, to support a variety of robotics accessories inside, including an A/V wireless surveillance system, radio Ethernet, onboard computer, laser range finder and more.

[2] A client command lets you engage/disengage the motors programmatically. See chapter 6.

On all models except those outfitted with the docking-charging system, a hinged rear door gives you easy access to the batteries, which you may quickly hot-swap to refresh any of up to three batteries.

Nose

The nose is where we put the optional onboard PC. The nose is readily removable for access: Simply remove two screws from underneath the front SONAR array. A third screw holds the nose to the bottom of the AT's body. The DX nose is hinged at the bottom.

Once the mounting screws are removed, simply pull the nose away from the body.[3] This provides a quick and easy way to get to the accessory boards and disk drive of the onboard PC, as well as to the SONAR gain adjustment for the front SONAR array. The nose also is an ideal place for you to attach your own custom accessories and sensors.

Access Panels

All DX's come with a removable right-side panel through which you may install accessory connectors and controls. A special side panel comes with the onboard PC option, for example, which provides connectors for a monitor, keyboard, mouse and 10Base-T Ethernet, as well as the means to reset and switch power for the onboard computer.

AT's come with a single access panel in the deck. Fastened down with finger-tight screws, the User Control Panel and onboard computer controls are accessible beneath the hinged door.

All models come with an access port near the center of the deck through which to run cables to the internal components.

SONAR

Pioneer 3 supports up to four SONAR arrays, each with up to eight transducers that provide object detection and range information for collision avoidance, features recognition, localization, and navigation. The SONAR positions in all Pioneer 3 SONAR arrays are fixed: one on each side, and six facing outward at 20-degree intervals.

Multiplexed Operation

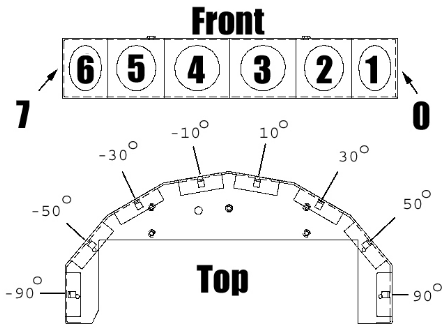
Each SONAR array's transducers are multiplexed: Only one disc per array is active at a time, but all four arrays fire one transducer simultaneously. The SONAR ranging acquisition rate is adjustable, normally set to 25 Hz (40 milliseconds per transducer per array). Sensitivity ranges from 10 centimeters (six inches) to five meters, depending on the ranging rate. You may control the SONAR's firing pattern through software, too; the default is left-to-right in sequence for each array. See the *ARCOS* Chapters 6 and 7 for details.

Sensitivity Adjustment

The driver electronics for each array is calibrated at the factory. However, you may adjust the array's sensitivity and range to accommodate differing operating environments. The SONAR gain control is on the underside of the SONAR driver board, which is attached to the floor of each SONAR module.

SONAR sensitivity adjustment controls are accessible directly, although you may need to remove the Gripper to access the front SONAR, if you have that accessory attached. For the front SONAR, for instance, locate a hole near the front underside of the array through which you can see the cap of the SONAR-gain adjustment potentiometer. Using a small flat-bladed screwdriver, turn the gain control counterclockwise to make the SONAR less sensitive to external noise and false echoes.

Low SONAR-gain settings reduce the robot's ability to see small objects. Under some circumstances, that is desirable. For instance, attenuate the SONAR if you are operating in a noisy environment or on uneven or highly reflective floor—a heavy shag carpet, for example. If the SONAR



Courtesy of *Activ*Media Robotics, LLC

Figure 6. Pioneer 3 SONAR array

[3] With older Pioneer 2 models, you also needed to remove the Gripper before removing the nose. With P3 models, the robot's nose and Gripper come off together, so you only need to remove the nose mounting screws.

are too sensitive, they will "see" the carpet immediately ahead of the robot as an obstacle.

Increase the sensitivity of the SONAR by turning the gain-adjustment screw clockwise, making them more likely to see small objects or objects at a greater distance. For instance, increase the gain if you are operating in a relatively quiet and open environment with a smooth floor surface.

MOTORS, WHEELS, AND POSITION ENCODERS

Pioneer 3 drive systems use high-speed, high-torque, reversible-DC motors, each equipped with a high-resolution optical quadrature shaft encoder for precise position and speed sensing and advanced dead-reckoning. Motor gearhead ratios, encoder ticks-per-revolution and tire sizes vary by robot model. However, ARCOS can correct for tire mismatches and convert most client commands and reported server information from platform-independent distance and heading units into platform-dependent encoder ticks, as expressed in the `DriftFactor, TicksMM` and `RevCount` FLASH parameters. Please read Chapter 6 for more details.

All Pioneer 3-DX robots come with foam-filled solid tires with knobby treads.[4] Pioneer 3-AT tires are pneumatic so that you may configure your robot for differing terrains. In any configuration, be careful to inflate the 3-AT tires evenly and adjust the respective `DriftFactor`, `TicksMM` and `RevCount` FLASH parameters for proper operation. We ship Pioneer 3-AT's with the tires inflated to 23 psi each.

BATTERIES AND POWER

Pioneer 3 robots contain up to three hot-swappable 12V sealed lead-acid batteries, accessible through a hinged and latched rear door. We provide a suction cup tool to help grab and slide each battery out of its bay. Spring contacts inside the robot make contact with the battery terminal tabs when inserted.

Warning: if any spring contacts do not make full contact with the battery terminal tabs, then robot runtime will be reduced, and the other batteries in the robot are at risk of being over-charged during recharging.

Note: Batteries have a significant impact on the balance and operation of your robot. Under most conditions, we recommend operating with three batteries. Otherwise, a single battery should be mounted in the center, or two batteries inserted on each side of the battery container.

Balance the batteries in your robot.

Battery Indicators and Low Voltage Conditions

The User Control Panel contains a LED labeled `BATTERY` that indicates current battery voltage. Approximately 12.5 volts and above, the LED glows bright green. As the available battery voltage drops as it is used, the LED turns progressively orange and then red at approximately 11.5 volts, the lowest recommended level of discharge.

User Control Panel's buzzer, if active (see the ARCOS `SoundTog` client command and FLASH parameter), will sound a repetitive alarm if the battery voltage drops consistently below the `LowBattery` parameter (default 11.5 V DC) level. If the battery voltage drops below the `ShutdownVolts parameter` (default 11 V DC) the microcontroller automatically shuts down any active client connection. Low available battery voltage may also also cause accessory devices including the optional onboard PC to shut down.

Recharging

Typical battery recharge time using the recommended accessory charger varies according to the discharge state; it is roughly equal to three hours per volt per battery. The Power Cube accessory allows simultaneous recharge of three swappable batteries outside the robot.

With the high-speed (4A maximum current) charger, recharge time is greatly reduced. It also supplies sufficient current to continuously operate the robot and onboard accessories, such as the onboard PC and radios. But with the higher-current charger, care must be taken to charge at least two batteries at once. A single battery may overcharge and thereby damage both itself and the robot.

[4] A ribbed-tread tire is available optionally. Contact *MobileRobots* sales for details.

All our recommended chargers are specifically designed for safe lead-acid battery recharging. Indicators on the module's face show fast-charge mode (typically an orange LED) in which the discharged batteries are given the maximal current, and trickle mode (green LED indicator), which the batteries are given only enough current to remain at full charge.

Safety ARCOS Watchdogs

ARCOS contains a communications `WatchDog` that will halt the robot's motion if communications between a PC client and the robot server are disrupted for a set time interval. The robot will automatically resume activity, including motion, as soon as communications are restored.

ARCOS also contains a stall monitor. If the drive exerts a PWM drive signal that equals or exceeds a configurable level (`StallVal`) and the wheels fail to turn, motor power is cut off for a configurable amount of time (`StallWait`). ARCOS also notifies the client which motor is stalled. When the `StallWait` time elapses, motor power automatically switches back on and motion continues under client control.

You may reconfigure the various FLASH-based parameter values to suit your application. See Chapter 7, *Updating & Reconfiguring ARCOS*, for details.

Chapter 3 Accessories

Pioneer 3 robots have many accessory options. For convenience, we include a description of the more commonly integrated accessories in this document. Please also refer to the detailed documents that come with the accessory.

Joystick and Joydrive Mode

Although not all models come standard with an external joystick connector, your Pioneer 3 robot's microcontroller has a joystick port and ARCOS contains a `JoyDrive` server for manual operation.

Start driving your robot with a joystick any time when it is not connected with a client software program. Simply plug it into the joystick port and press the "fire" button to engage the motors.

To drive your robot with a joystick while it is connected with an ARIA client (overrides client-based drive commands for manual operation), you must have the client software send the ARCOS command #47 with an integer argument of one to enable the ARCOS joystick servers. Have your client send the ARCOS `JOYDRIVE` command #47 with an integer argument of zero to disable the joystick drive override.

The joystick's fire button acts as the "deadman"—press it to start driving; release it to stop the robot's motors. The robot should drive forward and reverse, and turn left or right in response and at speeds relative to the joystick's position.

While driving forward, pull back on the joystick into full-reverse to decelerate faster than normal.

When not connected with a client control program, releasing the joystick fire button stops the robot. However when connected with a client, the client program resumes automatic operation of your robot's drive system. So, for example, your robot may speed up or slow down and turn, depending on the actions of your client program.

You may adjust the maximum translation and rotation speeds and even disable JoyDrive mode, through special ARCOS FLASH configuration parameters. See Chapter 7, *Updating & Reconfiguring ARCOS*, for details.

Bumpers

Bump rings fore and aft provide contact sensing for when other sensing has failed to detect an obstacle. The accessory rings also are segmented for contact positioning.

Electronically and programmatically, the bumpers trigger digital events which are reflected in the `STALL` values of the standard server-information packet that ARCOS automatically sends to a connected client. Your client also may request a special `IOpac` server information packet that contains additional, more-detailed bumper, stall, and other I/O related information.

Your robot may not move if you unplug one or both bumpers.

ARCOS itself monitors and responds to protection triggers. For example, ARCOS' `bumpStall` server triggers a stall in the robot whenever one or more bumper segments get triggered while the robot is moving in the same direction (front forward or rear reverse). Please consult the *Appendix A* for interface details and the chapters on ARCOS, particularly the section which describes the contents of the `IOpac` server information packet, later in this manual for configuration and programming details.

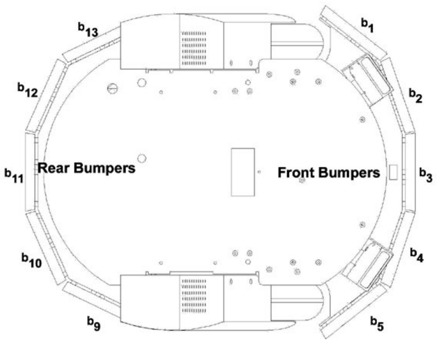


Figure 7. Pioneer 3 bumpers and associated STALL bits

Automated Recharging Accessory

The Pioneer 3-DX automated recharging accessory[5] is both a manual and an automated mechanism. Onboard controls, triggered either by the DEPLOY CHARGER button near the manual CHARGE port or by ARCOS-mediated client commands, deploy actuated contacts on the bottom of the robot, which in turn seat onto the power platform. Then, when activated by an IR-based, unique frequency-modulated signal from the robot, the power platform delivers up to 17 VDC @ 11.5 A to its plates.

Manual Operation (Robot Power OFF)

With MAIN POWER off, place the robot over the power platform so that its charging contacts are perpendicular to and, when deployed, contact the charger plates. Note that no charging power is applied to the plates on the platform; only low signal (5VDC @ <300mA) power for the IR detectors.

Press and hold the DEPLOY CHARGER button to manually deploy the power-contact mechanism on the bottom of the robot. Hold for a few seconds, but not more than 10 seconds. Charging is activated by positive contact with the power platform. In that case, the charge lamp on the power unit will light and the robot's contacts will remain deployed when you release the DEPLOY CHARGER button. Otherwise, the mechanism will retract. In that case, re-position the robot and try again.

The robot's power-contact mechanism automatically retracts if you press the DEPLOY CHARGER button while charging, if you move the robot on the power platform and lose positive charging contact, or if you remove power from the power unit. In all cases, charging power is removed immediately from the power platform when not actively engaged by the robot.

Manual Operation (Robot Power and Systems ON)

Because the automated recharging system’s integrated circuitry actively adjusts to system loads, it can run your robot's onboard systems while properly and optimally recharging its batteries. And because the charging mechanism may be operated independently of your robot's systems power, you may start up and shut down your robot and its onboard systems without disturbing the battery charging cycle, if engaged.

For example, with MAIN POWER on, use JoyDrive mode to position the robot onto the power platform. Then manually deploy the power-contact mechanism as described in the section above. Thereafter, switch MAIN POWER off, or conversely, start up and shut down other onboard systems, including the PC, camera, laser and other accessories, to proceed with development work without disturbing battery recharging.

The same conditions apply to remove power and retract the robot's power-contact mechanism with the robot’s MAIN POWER on as well as off. Since the ARCOS microcontroller always is active while the robot’s power is on, you also may connect and disconnect a client program, run in maintenance mode, or engage JoyDrive mode. However, engaging the motors, such as when you press the “fire” button on the joystick, immediately and automatically removes charging power and retracts the power-contact mechanism. And the mechanism will not activate manually via the DEPLOY CHARGER button until you disengage the motors.

Radio Controls and Accessories

All Pioneer platforms are servers in a client-server architecture. You supply the client software to run your intelligent mobile-robot applications. The client can run on either an onboard piggy-back laptop or embedded PC, or an off-board PC connected through radio modems or wireless serial Ethernet. In all cases, that client PC must connect to the internal HOST or User Control Panel SERIAL port in order for the robot and your software to work.

[5] The power-contact mechanism and onboard power conditioning circuitry can be retrofitted to all Pioneer 3 and some Pioneer 2 and Performance PeopleBot robots. All require return to the factory.

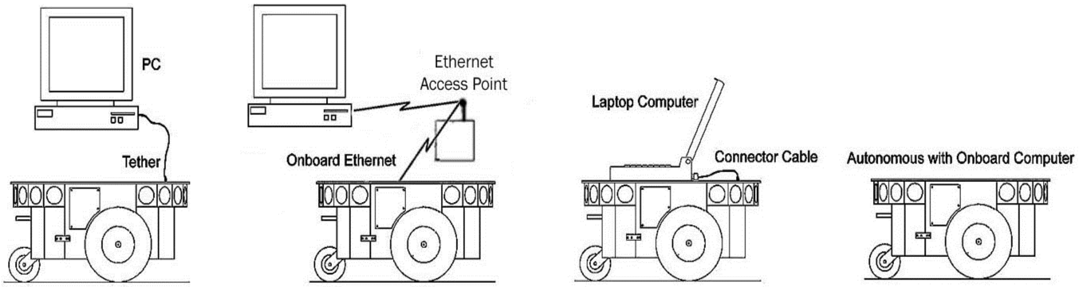


Figure 8. Client-server connection options.

For a piggyback laptop or the embedded PC, the serial connection is via a common "pass-through" RS-232 serial cable.

We also offer a wireless Ethernet-to-serial bridge accessory which connects directly to your robot's microcontroller. It works by automatically translating network-based Ethernet packet communications into streaming serial for the robot microcontroller and back again.

Running your robot through wireless Ethernet to an onboard computer is different than with the Ethernet-to-serial bridge device. In the first case, you run your robot client software on the onboard PC and use wireless Ethernet to monitor and control that PC's operation. In the latter case, you run the client software on a remote PC.

Accordingly, a major disadvantage of the wireless Ethernet-to-serial device is that it requires a consistent wireless connection with the robot. Disruption of the radio signal—a common occurrence in even the most modern installations—leads to poor robot performance and very short ranges of operation.

This is why we recommend onboard client PCs for wider, much more robust areas of autonomous operation, particularly when equipped with their own wireless Ethernet. In this configuration, you run the client software and its interactions with the robot microcontroller locally and simply rely on the wireless connection to export and operate the client controls. Moreover, the onboard PC is often needed for local processing, such as to support a laser range finder or to capture and process live video.

INTEGRATED PC

Mounted just behind the nose of the robot, the Pioneer 3 integrated PC is a common EBX form-factor board that comes with up to four serial ports, 10/100Base-T Ethernet, monitor, keyboard and mouse ports, USB ports and a hard-disk. For additional functionality, such as video framegrabber, Firewire or wireless Ethernet interfaces, the onboard PC accepts PC104 and PC104-plus PCI interface cards that stack on the motherboard.

Computer power comes from a dedicated DC:DC converter, mounted nearby. The hard-disk drive is mounted to the robot's nose, in between a cooling fan and computer speaker.

The onboard PC communicates with the robot's microcontroller through the `COM1` serial port under Windows or `/dev/ttyS0` on Linux systems. The microcontroller automatically switches to this connection when software opens the serial port. Therefore, the `SERIAL` port on the external User Control Panel can be used for robot control if no software is running on the onboard computer.

The serial port is also used to indicate some aspects of robot microcontroller state: DSR (pin 4) is RS-232 high when the microcontroller operates normally; otherwise it is low when reset or in maintenance mode. Similarly, RING (pin 9) normally is low and goes RS-232 high when the robot's batteries drop below the value of the `ShutdownVolts` parameter in ARCOS configuration.

Computer Control Panel

User-accessible communication and control port connectors, switches and indicators for the onboard PC are on the Computer Control Panel, found on the right side panel of the DX or in the hinged control well next to the user

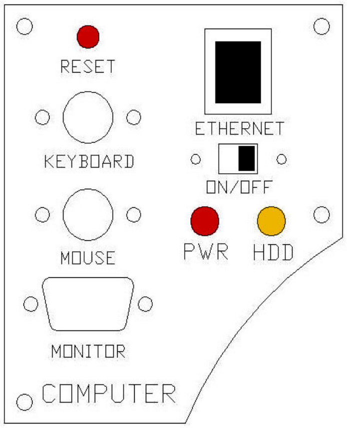


Figure 9. DX computer control side panel

controls of the AT. The controls and ports use common connectors: standard monitor DSUB and PS/2 (older computers) or USB (newer computers) connectors on the mouse and keyboard. The Ethernet is a 10/100Base-T standard RJ-45 socket.

The `ON/OFF` slide switch controls power to the onboard PC. The `PWR` LED lights when the computer has power.

The `HDD` LED lights when the onboard hard-disk drive is active. The `RESET` button restarts the PC.

Operating the Onboard PC

MobileRobots software is installed in `/usr/local` on Linux systems or in `C:\Program Files\MobileRobots` under Windows. In addition, we install the appropriate drivers for all computer devices included such as image framegrabber, wireless ethernet, sound and video.

Switch Main Power and Computer Power on. After boot up, log in to the system.

On Linux, two user accounts have been created: a normal user named 'guest' and the administrative account, which is named 'root'. The default passwords for both accounts is mobilerobots. We recommend changing these passwords using the passwd command.

On Windows, the administrative account is named 'Administrator'. The default password is 'mobilerobots'.

To log in remotely to a Linux system over a wireless network, use ssh (from Windows, use the free Putty application, available online). To log in remotely to a Windows system, use Remote Desktop.

To transfer files to a Linux system, use scp (from Windows, use the free WinSCP application, available online).

As a security measure, Linux normally prevents remote login as `root`; you must log in as a normal user and use the su and sudo commands to perform tasks that require root permissions.

Linux has been configured to normally boot into a non-graphical mode for improved performance. To access the graphical environment instead, choose the graphical UI mode at boot or use the '`startx`' command to start X-Windows and the GUI.

Please consult the computer manual and additional resources available at http://robots.mobilerobots.com for additional information on the onboard computer.

Onboard PC Networking

The RJ-45 connector on the Computer Control Panel provides wired 10/100Base-T Ethernet networking directly with the onboard PC. With the purchased option, we also install a wireless ethernet interface on the PC. The wireless Ethernet antenna sits atop the robot's deck.

Computer networking has been preconfigured by default with a static IP address with Class-C network configuration. The default network settings are:

- wired ethernet: Static IP address 10.0.125.32 (older robots had 192.168.100.32 instead).
- wireless network: Managed (not ad-hoc) mode with access point ESSID "Wireless Network", Static IP address 10.0.126.32 (192.168.1.32 on older robots).

If you also purchased a wireless access point from MobileRobots, then these default settings will work with that access point.

If instead, you are planning on using the robot on an existing wireless network, you will need to change these settings. In particular, you may need to use dhcp instead of a static IP address, or change the static IP address subnet, and change the wireless network ESSID and set the WEP key if used.

On Debian Linux, network settings are changed by editing the /etc/network/interfaces file or by using the Network settings control panel in the X-Windows GUI environment (System->Preferences). Additional information is available at http://robots.mobilerobots.com/wiki/Debian_Linux_Network_Configuration.

On Windows, network settings are changed in the network settings control panel. See http://robots.mobilerobots.com/wiki/Windows_Network_Configuration

Some robot devices use the ethernet network for communication, such as the SICK LMS-100 laser rangefinder, or an ethernet camera. These will either be connected to the second ethernet interface on the onboard computer, or to a small ethernet switch in the robot. These devices are on different logical networks and so virtual interfaces are used in the computer OS to communicate with them. Unless you change internal settings on these devices, you can leave them as configured by MobileRobots.

GYROSCOPE

The accessory gyroscope provides maximum 300 degrees-per-second rotational rate data to the AN6 analog port on your robot's microcontroller. With all versions, ARCOS and ARIA-based client software provide client-side support for the gyro. With the FLASH parameter `HasGyro` set to 1, the microcontroller, upon receiving the client command `GyroRequest` (#58 with argument 1), will acquire, average, and relay rate data (GYROpac 0x98) to ARIA, which applies the data to correct and compute the robot's actual x,y, th position, relieving errors due to wheel slippage, for instance. The gyro's calibration setting for this mode of operation is found as the `GyroScaler` value in your robot's respective ARIA parameters (".p") file; `p3dx-sh.p`, for example.

In ARCOS version 2.0 and later, there is server-side support for the gyro with finer heading resolution and more reliable calibration. The new `GyroCW` and `GyroCCW` FLASH parameters, with companion `CYROCALCW` (#38) and `GYROCALCCW` (#39) client commands, let you set the clockwise and counterclockwise configuration values separately (they tend to be different) for computing heading. The FLASH-based values are more permanently associated with the platform and much less likely to be changed or moved, as often happens with the mostly platform independent ARIA client software.

Set `HasGyro` to 2 in order to have ARCOS automatically use the gyro rate data to compute the `ThPos` heading and XPos, YPos position values in the standard SIP, as well as automatically correct for platform misalignment due to encoder-based rotations, such as if you manually rotate the robot without moving its wheels.

In all cases, the translation component for wheel velocities and position integration, and the rotational velocity, are encoder-based. And, of course, you may choose to ignore the gyro and rely on entirely on the encoders for your platform odometry. Consult the ARCOS Chapters 6 and 7 for more details.

Chapter 4 Quick Start

PREPARATIVE HARDWARE ASSEMBLY

Your Pioneer 3 robot comes fully assembled and ready for out-of-the-box operation. However, you may need to attach some accessories that were shipped separately for safety. The procedures we describe herein are for control of the basic robot.

The first time you operate and test the robot, we recommend that you put the robot up on blocks so that it cannot inadvertently.

If you have the onboard PC option, attach a keyboard, monitor, and mouse to the Computer Control Panel. Switch the computer power switch to ON.

CAREFUL
Slide the batteries into the robot TERMINALS LAST.
Otherwise, you will damage the robot.

Install Batteries

Out of the box, your Pioneer 3 robot comes with its batteries fully charged. The batteries are shipped separately.. Slide at least one and up to three batteries into the robot's battery box through the back door. : If using a single battery, place it in the center. If using two batteries, place one at either side. If using three batteries, use all three locations.

Make sure that all battery terminals are making contact with the spring contacts in the robot. If any batteries fail to make full contact, then damage may result to the batteries or robot.

CAREFUL
Always ensure that battery terminals are making full contact with the robot's spring contacts.

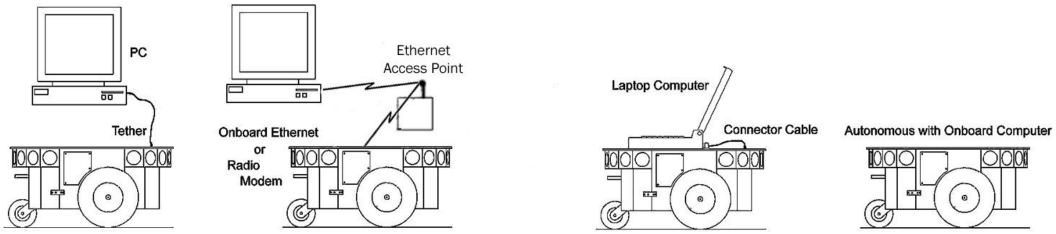
Client-Server Communications

Your robot requires a serial communication link with a client PC for operation. The serial link may be:

- ✓ A tether cable from the robot's 9-pin serial connector on the User Control Panel to a computer
- ✓ A piggyback laptop cabled to the User Control Panel serial port
- ✓ Serial Ethernet
- ✓ Radio Modem
- ✓ An integrated onboard PC wired internally for direct onboard control

ARIA DEMO

ARIA's examples are simple text-based programs that show how to make a connection to the robot controller and any required accessory devices, and then demonstrate how various ARIA, robot and accessory features can be used in your own software.



Communication between the PC running client software and the robot microcontroller is RS-232 serial at 9600 baud.

If using the optional onboard computer, this connection is `COM1` (Windows) or `/dev/ttyS0` (Linux) at 9600 baud.

To connect a laptop or other user computer through the HOST or SERIAL port on the side of the robot, use a serial cable or RS-232 compatible USB-serial adapter. If your computer uses a different device name for this connection, make sure to connect using this device name. (For example, a higher COM port on Windows or /dev/ttyUSB0 on Linux when using a USB adapter.) The connection port can be specified on the command line when using the ARIA library (see below and ARIA documentation.)

If your robot has a “WiBox” wireless Ethernet-to-serial device, you can run software on a remote computer, and configure it via ARIA command-line options to communicate with the WiBox. See the additional documentation provided with the WiBox for details.

Windows users may select the ARIA `demo` from the `Start` menu, in the `MobileRobots` program group, or open a Windows Command Prompt, enter the ARIA directory, and run the program:

```
cd “C:\Program Files\MobileRobots\ARIA”
```

`If using the 32-bit version of ARIA:`

```
cd bin
.\demo.exe
```

If using the 64-bit version of ARIA:

```
cd bin64
.\demo.exe
```

Linux users will find ARIA in `/usr/local/Aria/` :

```
cd /usr/local/Aria/bin
./demo
```

Demo Startup Options

By default, the ARIA `demo` program attempts to connect with the Pioneer 3 robot controller through the serial port `COM1` if on Windows or `/dev/ttyS0` if on Linux. To change connection options, command-line arguments may be used to set options (see Table 1.)

For example, from the Windows `Start:Run` dialog, choose `Browse…` and select the ARIA demo program: `C:\Program Files\MobileRobots\ARIA\bin\demo.exe`. Then, type a command line argument at the end of the text in the Run dialog. To connect through the Ethernet-to-serial radio device over the wireless network, for example, try the command:

```
C:\Program Files\MobileRobots\ARIA\bin\demo.exe -remoteHost 10.0.126.11
```

(Replace 10.0.126.11 with the correct IP address for your wireless ethernet-serial bridge device if different. E.g. some older devices use 192.168.1.32 instead by default.)

You can add additional options. Run `.\demo.exe -help` or see `CommandLineOptions.txt`. See ARIA reference documentation for more information on command line options.

Table 1. ARIA demo's robot connection start-up options

-remoteHost <Host Name or IP> (abbreviated -rh)	Connect with robot through a remote host over the network instead of a serial port; requires special serial Ethernet hardware or IPTHRU software mediation.
-robotPort <Serial Port> (abbreviated -rp)	Connect with robot through specified serial port name; COM3, for example. COM1 or /dev/ttyS0 is the default.
-robotBaud <baudrate> (abbreviated -rb)	Connect with robot using the specified baudrate; 19200 or 38400, for example. Default is 9600.
-remoteRobotTcpPort <Number> (abbreviated -rrtp)	Remote TCP host-to-robot connection port number; default is 8101.

A Successful Connection

ARIA prints out lots of diagnostic log messages as it negotiates a connection with the robot. If successful, the client requests various ARCOS services to start their activities, including SONAR polling, position integration and so on. The microcontroller sounds an audible connection cue and you should hear the robot's SONAR ping with a distinctive and repetitive clicking. In addition, the motors-associated STATUS LED on the User Control Panel should flash very fast (was flashing slowly while awaiting connection). Note that the ARIA demo automatically engages your robot's motors though a special client command. Normally, the motors are disengaged when first connecting.

The amber SERIAL port indicator LEDs on the robot's User Control Panel should blink to indicate ARIA-client to ARCOS-server communications, too.

Table 2. Keyboard teleoperation

KEY	ACTION
↑	forward
↓	reverse
←	turn left
→	turn right
space	all stop

Table 3. ARIA demo operation modes

MODE	HOT KEY	DESCRIPTION
laser	l	Displays the closest and furthest readings from the laser range finder
io	i	Displays the digital and analog-to-digital I/O ports
position	p	Displays the coordinates of the robot's position relative to its starting location
bumps	b	Displays bumpers status
SONAR	s	Displays the SONAR readings
camera	c	Controls and exercises the pan-tilt-zoom robotic camera
gripper	g	Controls, exercises and displays status of the Gripper accessory
wander	w	Sends the robot to move around at its own whim while avoiding obstacles
teleop	t	Drive and steer the robot via the keyboard or a joystick; avoids collisions
unguarded	u	Same as teleop, except no collision avoidance
direct	d	Direct command mode

Ir

OPERATING THE ARIA DEMONSTRATION CLIENT

When connected with the ARIA demo client, your robot becomes responsive and intelligent. For example, it moves cautiously. Although it may drive toward an obstacle, your robot will not crash because the ARIA demo includes obstacle-avoidance behaviors which enable the robot to detect and actively avoid collisions.

The ARIA demo displays a menu of robot operation options. The default mode of operation is teleop. teleop mode, you drive the robot manually, using the arrow keys on your keyboard or a joystick connected to the client PC's joystick port (as opposed to a joystick port on the robot).

While driving from the keyboard, hold down the respective keys to simultaneously drive the robot forward or backward and turn right or left. For instance, hold down the up-arrow key to have the robot accelerate forward to its cruising speed of around 400 millimeters per second (defined in the source code). Release the arrow key to have the robot slow down and stop. Press and hold the right- or left-arrow key to have the robot rotate or turn in an arc if you also hold down the up- or down-arrow key.

The other modes of ARIA demo operation give you access to your robot's various sensors and accessories, including encoders, SONAR, laser, Gripper, a pan-tilt-zoom robotic camera, I/O port states, bumpers and more. Accor-dingly, use the ARIA demo not only as a demonstration tool, but as a diagnostic one, as well, if you suspect a sensor or effector has failed or is working poorly. The demo also is useful for calibrating your robot's drive system.

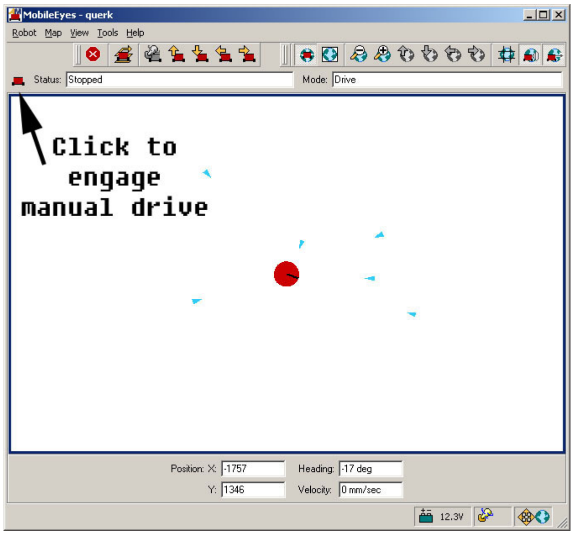
Access each ARIA demo mode by pressing its related hot-key: '`t`', for instance, to select `teleoperation`. Each mode includes onscreen instructions and may have sub-menus for operating of the respective device.

Disconnecting

When you finish, press the `Esc` key to disconnect the ARIA client from your robot server and exit the ARIA demonstration program. Your robot should disengage its drive motors and stop moving, and its SONAR should stop firing. You may now slide the robot's `Main Power` switch to `OFF`.

Networking with MobileEyes

To use the MobileEyes GUI client for much more advanced, network-based robotics control, you need to do things a bit differently. You need the Ethernet-to-serial device or a PC on the robot. But instead of ARIA `demo`, use ArNetworking's `serverDemo` to mediate communications between MobileEyes and your robot over the network.



Status: Stopped
Mode: Drive

Click to engage manual drive

Position: X: -1757
Y: 1346
Heading: -17 deg
Velocity: 0 mm/sec
12.3V

Figure 10. MobileEyes connected to the basic ArNetworking serverDemo

Start serverDemo

The ArNetworking client, `serverDemo`, works to mediate communications between your Pioneer 3 robot over the network and remote clients, such as the GUI MobileEyes. To examine its inner workings, look at `serverDemo.cpp` (and others) sources in the `ArNetworking/examples` subdirectory.

Start `serverDemo` just like `demo`, with the `–remoteHost` argument if you aren't running the software onboard. ServerDemo also accepts the special command-line argument –connectLaser if you have a SICK LMS rangefinder attached to the serverDemo-based PC. ServerDemo can be run anywhere on the network from which your MobileEyes and your robot have access, such as on the same PC as you run MobileEyes.

Connect MobileEyes with serverDemo, not the robot's microcontroller.

Start MobileEyes and Connect with serverDemo

From Windows, simply double-click the MobileEyes.exe program normally located in the `C:\Program Files\MobileRobots\MobileEyes\bin` directory. With Linux, you need to have started X and, from a terminal session, navigate to the `MobileEyes/bin` directory in `/usr/local` and execute it:

```
./MobileEyes &
```

In the MobileEyes startup dialog, enter the hostname or IP address of the PC on which you are running `serverDemo`.

Operating MobileEyes

Simply click the robot icon on the menu bar to engage manual drive and teleoperate the robot. Your robot with MobileEyes really comes to life when used with SONARNL or ARNL for map-based path-planning and navigation.

QUICKSTART TROUBLESHOOTING

Most problems occur when attempting to connect the ARIA or ArNetworking demo program with a robot for the first time. The process can be daunting if you don't make the right connections and installations.

Parameter Files

The most common mistake is not having your robot's "parameter" file located in the `Aria/params` directory. For Pioneer 3's, the ARIA-parameter filenames are `p3dx-sh.p or p3at-sh.p`. These and all other ARIA parameter files either come with the latest version of. Note that the ARIA library itself contains internal parameter defaults in case the parameter file is missing. And you can make up your own custom robot parameter files, too, using the value of the robot's `NAME` parameter.

Proper Connections

Make sure you have the software properly installed and that your robot and connections are correct. A common mistake with Linux is not having the proper permissions on the connecting serial port.

Make sure your robot's batteries are fully charged (battery LED green). The robot servers shut down and won't allow a connection at under `ShutdownVolts`.

ATTENTION!

The demo/serverDemo-to-robot connection is *SERIAL only*. Accordingly, run them on the onboard or piggyback computer, over radio modems or over the network with a wireless Ethernet-to-serial device.

If you are using the onboard PC or radios, the serial connection is internal and established at the factory; you should not have problems with those cables. Simply make sure the `AUX1` switch on the User Control Panel is engaged (associated LED lit), for example. And remove any serial cable that is plugged into the User Control Panel as it may interfere with internal serial communication.

With other serial connections, make sure to use the proper cable: a "pass-through" one, minimally connecting pins 2, 3, and 5 of your PC's serial port to their respective contacts of the robot's serial port on the User Control Panel.

If you access the wrong serial port, the demonstration program will display an error message. If the robot server isn't listening or if the serial link is severed somewhere between the client and server (cable loose or the radio is off, for instance), the client will attempt "Syncing 0" several times and fail. In that case, `RESET` the robot and check your serial connections.

If for some reason communications get severed between the client and ARCOS server, but both the client and server remain active, you may revive the connection with little effort. If you are using wireless communications, first check and see if the robot is out of range.

Communications also will fail if the client and/or server is somehow disabled during a session. For instance, if you inadvertently switch off the robot's `Main Power` or press the `RESET` button, you must restart the connection. Turning the `Main Power` switch `OFF` and then back `ON`, or pressing the `RESET` button puts the robot servers back to their wait state, ready to accept client connections again. If the demo or other client application is still active, simply press `esc` and restart.

Chapter 5 Software and Robot Control

ROBOT CONNECTION

All Pioneer platforms use a two-tier client-server mobile robot-control architecture. In the model, firmware on the robot's embedded controller works to manage all the low-level details of the mobile robot's systems. These include operating the motors, firing the SONAR, collecting and reporting SONAR and wheel encoder data and so on—all on command from and reporting to a separate client application, such as the ARIA demo.

Communication between the optional onboard PC and the robot microcontroller is RS-232 serial through `COM1` (Windows) or `/dev/ttyS0` (Linux) at 9600 baud. To connect a laptop or other user computer through the HOST or SERIAL port on the side of the robot, use a serial cable or RS-232 compatible USB-serial adapter. If your computer uses a different device name for this connection, make sure to connect using this device name. (For example, a higher COM port on Windows or /dev/ttyUSB0 on Linux when using a USB adapter.) The connection port can be specified on the command line when using the ARIA library (see Chapter 5, Quick Start, above, and ARIA documentation.)

With this client/server architecture, robotics applications developers do not need to know many details about a particular robot server, because the client insulates them from this lowest level of control. Some of you, however, may want to write your own robotics control and reactive planning programs, or just would like to have a closer programming relationship with your robot. This chapter explains how to communicate with and control your robot via the Advanced Robot Control and Operations Software (ARCOS) client-server interface. The same ARCOS functions and commands are supported in the various client-programming environments that accompany your robot or are available for separate license.

ARCOS is compatible with all MobileRobots platforms, implementing the same commands and information packets that first appeared in the Pioneer 1-based PSOS, in the original Pioneer 2-based P2OS, and more recent AROS-based Pioneer 2s and 3s, as well as PeopleBot and PowerBot. ARCOS, of course, extends the servers to add new functionality, improve performance, and provide additional information about the robot's state and sensing.

ROBOTS IN MOTION

Robot motion is possible on two axes: translation (forward or backward on the local X axis, specified as mm/second) and rotation (clockwise or counter clockwise specified as degrees/second). Your client software can send requests for desired speed on each of these axes independently. ARCOS will control the robot drive wheels accordingly, resulting in your desired trajectory of the platform.

Alternatively, client software can request a discrete movement on either axis: movement along a specific distance forward or backward, or a rotation by a specified number of degrees, or to a specified target heading in the robot coordinate system (see below).

Requesting Robot Motion

Translational velocity can be requested via ARIA (ArRobot::setVel() or via an ArAction object), or using the VEL command. Rotational velocity can be requested via ARIA (ArRobot::setRotVel()) or via an ArAction object), or using the RVEL command.

When ARCOS receives a motion command, it accelerates or decelerates the robot to achieve the requested translation or rotation speed, or to a default speed used to achieve a distance movement. Translation acceleration or deceleration can be set via ARIA (ArRobot::setAccel(), ArRobot::setDecel()) or via SETA command #5. Rotational acceleration or deceleration can be set via ARIA (ArRobot::setRotAccel(), ArRobot::setRotDecel()) or via SETRA command #23.

Default speeds for distance movement can be set via SETV and SETRV commands.

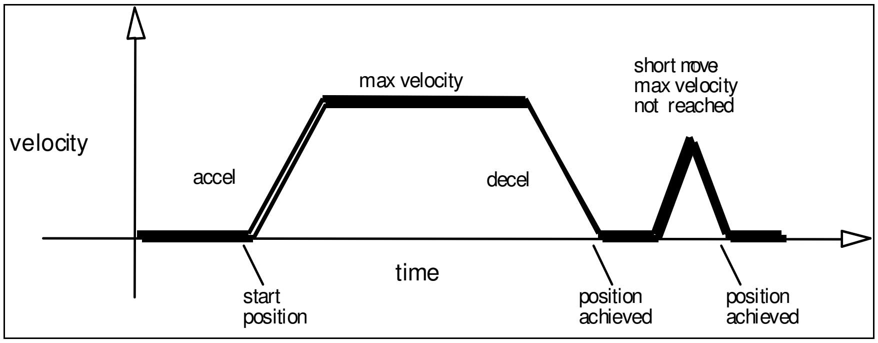


Figure 11. Trapezoidal velocity profile

Discrete distance movements can be requested via ARIA (ArRobot::move()) or the MOVE command. The MOVE command sends the platform forward the specified distance in millimeters if a positive argument, or backward if a negative distance. Each move is limited to +5000 millimeters and -4999 millimeters.

Discrete changes in orientation can be requested via ARIA (ArRobot::setHeading()) or the HEAD or DHEAD commands. HEAD rotates the robot to the specified heading in the global robot coordinate system. DHEAD changes it relative to its current heading. In general, positive relative heading command arguments turn the robot in a counterclockwise direction. However, the robot always turns in the direction that will achieve its heading most efficiently. Accordingly, relative-heading arguments greater than 180 degrees automatically get reduced to 180 or less degrees with a concomitant change in direction of rotation.[6]

The `E_STOP` command #55 or the `STOP` button that is found on some Pioneer robot platforms override deceleration and abruptly stop the robot in the shortest distance and time possible. Accordingly, the robot brakes to zero translation and rotation velocities with very high deceleration and remains stopped until it receives a subsequent translation or rotation velocity command from the client or until the STOP button is reset. (See *E_STOP and E_STALL* later in this chapter.)

PID Controls

The ARCOS drive servers use a common Proportional-Integral-Derivative (PID) system with wheel-encoder feedback to adjust a pulse-width-modulated (PWM) signal at the motor drivers to control the power to the motors. The motor-duty cycle is 50 microseconds (20 KHz); pulse-width is proportional 0-500 for 0-100% of the duty cycle. The ARCOS drive servers recalculate and adjust your robot's trajectory and speed every five milliseconds.

The PID values for translation and rotation and maximum PWM are reconfigurable FLASH parameters in your robot's microcontroller. You also may temporarily update the PID values with the ARCOS client commands #84 through #87. On-the-fly changes persist until the client disconnects. Translation PID values apply to independent wheel-velocity mode, as well.

The P-term value Kp increases the overall gain of the system by amplifying the position error. Large gains will have a tendency to overshoot the velocity goal; small gains will limit the overshoot but cause the system to become sluggish. We've found that a fully loaded robot works best with a Kp setting of around 15 to 30, whereas a lightly loaded robot may work best with Kp in the range of 20 to 50.

The D-term Kv provides a PID gain factor that is proportional to the output velocity. It has the greatest effect on system damping and minimizing oscillations within the drive system. The term usually is the first to be adjusted if you encounter unsatisfactory drive response. Typically, we find Kv to work best in the range of 10 to 30 for lightly to heavily loaded robots, respectively. If your robot starts to vibrate or shutter, reduce Kv.

The I-Term Ki moderates any steady state errors thereby limiting velocity fluctuations during the course of a move. At rest, your robot will seek to "zero out" any command position error. Too large of a Ki factor will cause an excessive windup of the motor when the load changes, such as when climbing over a bump or accelerating to a new speed.

[6] Also when operating the robot with the server-side gyroscope (`hasGyro 2`), the HEAD command gets repeated internally to ensure that the platform reaches its intended heading.

.

Consequently, we typically use a minimum value for Ki in the range of 0 to 10 for lightly to heavily loaded robots respectively.

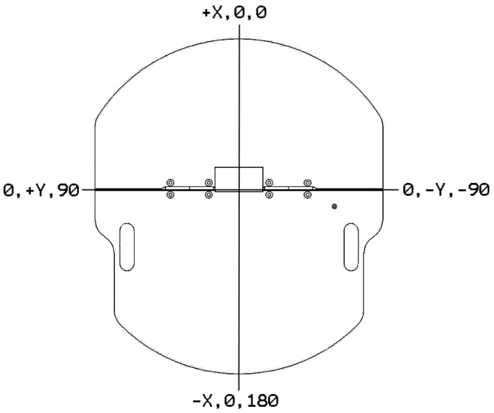


Figure 12. Internal coordinate system

Position Integration

Pioneer robots track their position and orientation based on dead-reckoning from wheel motion derived from encoder readings and from the integrated gyroscope accessory when attached and enabled. The ARCOS-based robot maintains its internal coordinate position in platform-dependent units, but reports the values in platform-independent millimeters and angular units (2π/4096 radians) in the standard SIP (Xpos, Ypos, and Thpos).

Be aware that registration between external and internal coordinates deteriorates rapidly with movement due to gearbox play, wheel imbalance and slippage, and many other real-world factors. You can rely on the dead-reckoning ability of the robot for just a short range—on the order of a few meters and one or two revolutions, depending on the surface. Carpets tend to be worse than hard floors.

Also, moving either too fast or too slow tends to exacerbate the absolute position errors. Accordingly, consider the robot's dead-reckoning capability as a means of tying together sensor readings taken over a short period of time, not as a method of keeping the robot on course with respect to a global map.

On start-up, the robot is at the origin (0, 0, 0), pointing along the positive X-axis at 0 degrees. Absolute angles vary between 0 and ±4096 angular units (+180 to -179 degrees).

You may reset the internal coordinates back to 0,0,0 with the `SETO` command #7.

CLIENT-SERVER COMMUNICATION PACKET PROTOCOLS

The Pioneer 3 communicates with a client using a client-server communication packet protocols, one for command packets from client to server and another for Server Information Packets (SIPs) from the server to client. Both protocols are byte streams consisting of five main elements: a two-byte header, a one-byte count of the number of subsequent packet bytes, the client command or SIP packet type, command data types and argument values or SIP data bytes, and, finally, a two-byte checksum. Packets are limited to a maximum of 207 bytes each.

The two-byte header which signals the start of a packet is the same for both client command packets and SIPs: 0xFA (250) followed by 0xFB (251). The subsequent count byte is the number of all *subsequent* bytes in the packet including the checksum, but not including the byte count value itself or the header bytes.

Data types are simple and depend on the command or specific packet (see descriptions below): client commands, SIP types, and so on, are single 8-bit bytes, for example. Command arguments and SIP values may be 2-byte integers, ordered as least-significant byte first. Some data are strings of up to a maximum 200 bytes, prefaced by a length byte.

The two bytes `c1` and `c2` of a 2-byte integer can be combined into an integer `i` like this:

```
i = (c1 & 0xff) | (c2 << 8);
```

Note that unlike data integers, however, the two-byte checksum at the end of each packet appears with its *most-significant* byte first (see below for example code).

Table 4. Client command packet protocol

COMPONENT	BYTES	VALUE	DESCRIPTION
header	2	0xFA, 0xFB	Packet header; same for client and server
byte count	1	N	Number of command/argument bytes plus Checksum’s two bytes, but not including Byte Count itself or the header bytes. Maximum of 249.
command number	1	0 – 255	Client command number; see Table 7.
argument type	1	0x3B or 0x1B or 0x2B	Required data type of command argument: positive integer, negative or absolute integer, or string
argument	n	data	Command argument; always 2-byte integer or string containing length prefix
checksum	2	computed	Packet integrity checksum

Packet Checksum

Calculate the client-server packet checksum by successively adding data byte pairs (high byte first) to a running checksum (initially zero), disregarding sign and overflow. If there are an odd number of data bytes, the last byte is XORed to the low-order byte of the checksum.

```
AREXPORT ArTypes::Byte2 ArRobotPacket::calcCheckSum(void)
{
  int i;
  unsigned char n;
  int c = 0;

  i = 3;
  n = myBuf[2] - 2;
  while (n > 1) {
    c += ((unsigned char)myBuf[i]<<8) | (unsigned char)myBuf[i+1];
    c = c & 0xffff;
    n -= 2;
    i += 2;
  }
  if (n > 0)
    c = c ^ (int)((unsigned char) myBuf[i]);
  return c;
}
```

(from MobileRobots ARIA ArRobotPacket.cpp)

NOTE: The checksum integer is placed at the end of the packet, with its bytes in the reverse order of that used for data; that is, b_0 is the high byte and b_1 is the low byte.

Packet Errors

ARCOS ignores a client command packet whose byte count exceeds 204 (total packet size of 207 bytes) or has an erroneous checksum. The client should similarly ignore erroneous SIPs.

Because of the real-time nature of client-server mobile-robotics interactions, we made a conscious decision to provide an unacknowledged communication packet interface. Retransmitting server information or command packets typically serves no useful purpose because old data is useless in maintaining responsive robot behaviors.

Nonetheless, the client-server interface provides a simple means for dealing with ignored command packets: Most of the client commands alter state variables in the server. By examining those values in respective SIPs, client software may detect ignored commands and re-issue them until achieving the correct state.

THE CLIENT-SERVER CONNECTION

Before exerting any control, a client application must first establish a connection with the robot server via a serial link through the robot microcontroller's HOST serial port either via the internal HOST or the User Control Panel SERIAL connector. After establishing the communication link, the client then sends commands to and receives operating information from the server.

When first started or reset, ARCOS is in a special wait state listening for communication packets to establish a client-server connection.[7] To establish a connection, the client application must send a series of three synchronization packets containing the SYNC0, SYNC1 and SYNC2 commands in succession, and retrieve the server responses.

Specifically, and as examples of the client command protocol described below, the sequence of synchronization bytes is:

```
SYNC0:  250, 251, 3, 0, 0, 0
SYNC1:  250, 251, 3, 1, 0, 1
SYNC2:  250, 251, 3, 2, 0, 2
```

When in wait mode, ARCOS echoes the packets verbatim back to the client. The client should listen for the returned packets and only issue the next synchronization packet after it has received the appropriate echo.

Autoconfiguration (SYNC2)

ARCOS automatically sends robot identifying information back to the client following the last synchronization packet (SYNC2). The configuration values are three NULL-terminated strings that comprise the robot's FLASH-stored name, type, and subtype. You may uniquely name your robot with the FLASH configuration tool we provide. The type and subtype are constants set at the factory and normally not changed thereafter. (See next chapter for details.)

The type string typically is Pioneer. The subtype depends on your robot model; P3DX-SH or P3AT-SH, for example. Clients may use these identifying strings to self-configure their own operating parameters. ARIA, for instance, loads and uses the robot's related parameter files found in the special Aria/params directory.

Opening the Servers—OPEN

Once you've established a connection with ARCOS, your client should send the OPEN command #1 (250, 251, 3, 1, 0, 1) to the server, which causes the microcontroller to perform a few housekeeping functions, start its various servers, such as for the SONAR and motors, and begin transmitting server information to the client.

Note that when at first connected, your robot's motors are disabled regardless of their state when last connected. To enable the motors after starting a connection, you must either do it manually (press the white MOTORS button on the User Control Panel), from the LCD accessory's interactive mode, or have your client send an ENABLE client command #4 with an integer argument of 1. See *Client Commands* below.

Once connected, send the ENABLE command
or press the white MOTORS button on the User Control Panel
to enable your robot's motors.

Server Information Packets

Once OPENed, ARCOS automatically and repeatedly sends a packet of information over the HOST serial port back to the connected client. The standard ARCOS SIP informs the client about a number of operating states and readings, using the order and data types described in the nearby Table. ARCOS also supports several additional SIP types. See following sections for details.

[7] There also is maintenance mode for ARCOS downloads and parameter updates; see next chapter for details.

Table 5. ARCOS standard SIP contents

LABEL	DATA	DESCRIPTION
HEADER	2 bytes	Exactly in order 0xFA (250), 0xFB (251)
BYTE COUNT	byte	Number of data bytes + 2 (checksum), not including header or byte-count bytes
TYPE	0x3s	Motors status; s = 2 when motors stopped or 3 when robot moving.
XPOS	int	Wheel-encoder and optional gyro integrated coordinates in millimeters (DistConvFactor‡ = 1.0).
YPOS	int	
THPOS	int	Orientation in angular units (AngleConvFactor‡ = 0.001534 radians per angular unit = 2π/4096).
L VEL	int	Wheel velocities in millimeters per second (VelConvFactor‡ = 1.0)
R VEL	int	
BATTERY	byte	Battery charge in tenths of volts (101 = 10.1 volts, for example)
STALL AND BUMPERS	uint╪	Motor stall and bumper indicators. Bit 0 is the left wheel stall indicator, set to 1 if stalled. Bits 1-7 correspond to the first bumper I/O digital input states (accessory dependent). Bit 8 is the right wheel stall, and bits 9-15 correspond with the second bumper I/O, also accessory and application dependent.
CONTROL	int	Setpoint of the server's angular position servo in degrees
FLAGS	uint	Bit 0 motors status; bits 1-4 SONAR array status; bits 5,6 STOP; bits 7,8 ledge-sense IRs; bit 9 joystick fire button; bit 10 auto—charger power-good.
COMPASS	byte	Electronic compass accessory heading in 2-degree units
SONAR COUNT	byte	Number of new SONAR readings included in SIP
NUMBER	byte	If SONAR Count>0, is SONAR disc number 0-31; readings follow
RANGE	uint	Corrected SONAR range value in millimeters (RangeConvFactor‡ = 1.0)
…REST OF THE SONAR READINGS…		
GRIP_STATE	byte	Gripper state byte.*
ANPORT	byte	Selected analog port number 1-5
ANALOG	byte	User Analog input (0-255=0-5 VDC) reading on selected port
DIGIN	byte	Byte-encoded User I/O digital input
DIGOUT	byte	Byte-encoded User I/O digital output
BATTERYX10	int	Actual battery voltage in 0.1 V (especially useful for battery voltages > 25.5)
CHARGESTATE	byte	Version 1.5 and later. Automated recharging state byte; -1 = unknown; 0=not charging; 1=bulk; 2=overcharge; 3=float.

ROTVEL	int	Current rotational velocity in degrees X 10 per sec
FAULTFLAGS	int	Not used on Pioneer.
LATVEL	int	Not used on Pioneer.
TEMPERATURE	byte	Not used on Pioneer.
STATEOFCHARGE	byte	Not used on Pioneer.
CHECKSUM	int	Packet-integrity checksum

‡ Client-side data-conversion factor. Consult the ARIA parameter file for your robot.

╪ Explicitly, an unsigned integer; all others sign-extended

Keeping the Beat—PULSE

An ARCOS safety watchdog expects that, once connected, your robot receives at least one command packet from the client every `watchDog` cycle, as defined in your robot's FLASH (default is two seconds). Otherwise, it assumes the client-server connection is broken and stops the robot.

Some clients—ARIA-based ones, for instance—use the good practice of sending a `PULSE` client command #0 (250, 251, 3, 0, 0, 0) just after `OPEN`. And if your client application will be otherwise distracted for some time, periodically issue the `PULSE` command to let your robot server know that your client is indeed alive and well. It has no other effect.

If the robot shuts down due to lack of communication with the client, it will revive upon receipt of a client command and automatically accelerate to the last-specified speed and heading setpoints.

Closing the Connection—CLOSE

To close the client-server connection, which automatically disables the motors and other functions like SONAR, simply issue the client `CLOSE` command #2 (250, 251, 3, 2, 0, 2).

Most of ARCOS' operating parameters return to their FLASH-based default values upon disconnection with the client.[8]

CLIENT COMMANDS

ARCOS has a structured command format for receiving and responding to directions from a client for control and operation of your robot. Client commands are comprised of a one-byte command number optionally followed, if required by the command, by a one-byte description of the argument type and then the argument value.

The number of client commands you may send per second depends on the `HOST` serial baud rate, average number of data bytes per command, synchronicity of the communication link, and so on. ARCOS' command processor runs on a one millisecond interrupt cycle, but the server response speed depends on the command. Typically, limit client commands to a maximum of one every 3-5 milliseconds or be prepared to recover from lost commands.

Argument Types

Table 6. Command Argument Types

int	One 16-bit signed integer value (LSB-first)
uint	One 16-bit unsigned integer value (LSB-first)
str	Variable-size buffer. If not otherwise specified, it is length-prefixed (first byte indicates number of bytes of data that follow). Some strings intended to hold printable text are NULL-terminated however. Read command description for details. Maximum, if not otherwise specified, is 200 bytes.
2-byte	Two signed 8-bit bytes.
none	No argument data follows.

[8] With earlier OSes, the changes persisted between sessions and reverted to the FLASH defaults only after the controller was reset.

Command Listing

Table 7. ARCOS client-side command set

COMMAND	#	ARGS	DESCRIPTION	VERSION
			Before Client Connection:	
SYNC0	0	none	Start connection. Send in sequence. ARCOS echoes	1.0
SYNC1	1	none	synchronization commands back to client, and	
SYNC2	2	none	robot-specific auto-synchronization after SYNC2.	
			After Established Connection:	
PULSE	0	none	Reset server watchdog.	1.0
OPEN	1	none	Start up servers.	1.0
CLOSE	2	none	Close servers and client connection.	1.0
POLLING	3	str	Change sonar polling sequence. See **SONAR** on page 32, below.	1.0
ENABLE	4	int	1=enable; 0=disable the motors.	1.0
SETA	5	int	Set translation acceleration, if positive, or deceleration, if negative; in mm/sec2.	1.0
SETV	6	int	Set maximum/move translation velocity; mm/sec.	1.0
SETO	7	none	Reset local position to 0,0,0 origin.	1.0
MOVE	8	int	Translate (+) forward or (-) back mm distance at SETV speed	1.0
ROTATE	9	int	Rotate (+) counter- or (-) clockwise degrees/sec at SETRV limited speed.	1.0
SETRV	10	int	Sets maximum/turn rotation velocity; degrees/sec.	1.0
VEL	11	int	Translate at mm/sec forward (+) or backward (-) (SETV limited speed).	1.0
HEAD	12	int	Turn at SETRV speed to absolute heading; ±degrees (+ = counterclockwise).	1.0
DHEAD	13	int	Turn at SETRV speed relative to current heading; (+) counter- or (-) clockwise degrees.	1.0
SAY	15	str	Play up to 20 tones duration, tone sound pairs through User Control Panel piezo speaker (buzzer). Provide length-prefixed buffer of (duration, tone) byte pairs. Duration in 20ms increments. Tone is MIDI note value. See http://robots.mobilerobots.com/wiki/SAY_Command for more information.	1.0
JOYREQUEST	17	int	Request one or continuous stream (>1) or stop (0) joystick SIPs	1.3
CONFIG	18	none	Request a configuration SIP. **See CONFIGpac and CONFIG Command** on page 34 below.	1.0
ENCODER	19	int	Request one, a continuous stream (>1), or stop (0) encoder SIPs. See **Encoders** on 36, below.	1.0
RVEL	21	int	Rotate robot at (+) counter- or (-) clockwise;	1.0

			degrees/sec (SETRV limit).	
DCHEAD	22	int	Adjust heading relative to last setpoint; ± degrees (+ = ccw)	1.0
SETRA	23	int	Change rotation de(-) or (+)acceleration, in degrees/sec2	1.0
SONAR	28	int	1=enable, 0=disable all the SONAR; otherwise, use bits 1-3 to specify an individual array number 1-4. See **SONAR** on page 32, below for more information.	1.0
STOP	29	none	Stop robot motion; cancel currently active motion commands. Motors remain enabled	1.0
DIGOUT	30	2 byte	Set (1) or reset (0) User Output ports. Bits 8-15 is a byte mask that selects, if set (1), the output port(s) for change; Bits 0-7 set (1) or reset (0) the selected port(s).	1.0
VEL2	32	2 byte	Set independent velocities for each side of the robot differential drive; First byte bits 0-7 for right wheel, second byte bits 8-15 for left wheel; in 20mm/sec increments. Note: accurate velocity control is not performed in this mode, the motors are simply commanded according to the requested linear velocities for each side. VEL and ROTVEL are suggested for more accurate automatic velocity control instead.	1.1
GRIPPER	33	int	Gripper server commands. See the Gripper or PeopleBot Manual for details.	1.0
ADSEL	35	int	Selects the A/D port number for reporting ANPORT value in standard SIP.	1.0
GRIPPERVAL	36	int	Gripper server values. See Gripper or PeopleBot Manual for details.	1.0
GRIPREQUEST	37	int	Request one, a continuous stream (>1), or stop (0) Gripper SIPs.	1.0
GYROCALCW	38	uint	Set the clockwise rotation calibration value for the gyro, used if HasGyro=2. (See **Heading Correction Gyro** on page 41 and **Updating & Reconfiguring ARCOS** on page 43 for details.)	2.0
GYROCALCCW	39	uint	Set the counterclockwise rotation calibration value for the gyro, used if HasGyro=2. (See **Heading Correction Gyro** on page 41 and **Updating & Reconfiguring ARCOS** on page 43 for details.)	2.0
IOREQUEST	40	int	Request one (1), a continuous stream (>1), or stop (0) IO SIPs.	1.0
TTY2	42	str	Send length-prefixed string argument to serial device connected to AUX1 serial port.	1.0
GETAUX	43	uint	Request to retrieve 1-200 bytes from the AUX1 serial port; 0 flushes the buffer.	1.0
BUMPSTALL	44	int	Stall robot if no (0), only front (1) while moving forward, only rear (2) while moving backward, or either (3) bumpers contacted when robot moving in related direction.	1.0
TCM2	45	int	TCM2 module commands; 0=module off, no readings; 1=compass only, readings in standard SIP; 2=send one TCM2 packet; 3=send continuous (each cycle) TCM2 packets; 4=user calibration; 5=auto calibration; 6=stop auto-calibration, send one packet, revert to mode 1; 7=soft reset	1.0

JOYDRIVE	47	int	1=allow joystick drive from port while connected with a client; 0 (default) disallows.	1.0
SONARCYCLE	48	uint	Change the sonar cycle time; in milliseconds. See **SONAR** on page 32, below.	1.0
HOSTBAUD	50	int	Change the HOST serial port baud rate to 0=9600, 1=19200, 2=38400, 3=57600, or 4=115200.	1.0
AUX1BAUD	51	int	Change the AUX1 serial port baud rate (see HOSTBAUD).	1.0
AUX2BAUD	52	int	Change the AUX2 serial port baud rate (see HOSTBAUD).	1.0
AUX3BAUD	53	int	Change the AUX3 serial port baud rate (see HOSTBAUD).	1.0
E_STOP	55	none	Emergency stop; very abrupt by overriding deceleration.	1.0
M_STALL	56	int	Argument 1=MOTORS button off causes a stall	1.0
GYROREQUEST	58	int	If client-side (HasGyro 1), request one, a continuous stream (>1), or stop (0) Gyro SIPs. If server-side (HasGyro 2), 0 disable or 1 to enable the gyro. (See **Heading Correction Gyro** on page 41 and **Updating & Reconfiguring ARCOS** on page 43.)	1.0
LCDWRITE	59	str	Display a message on the LCD accessory: byte 0=starting column (1-19); byte 1=starting row (1-4); byte 2=1 if clear line contents first, otherwise 0; bytes 3- = up to 20 chars, NULL-terminated.	1.3
TTY4	60	str	Send length-prefixed string argument out to device connected at AUX3 serial port.	1.0
GETAUX3	61	int	Request to retrieve 1-200 bytes from the device connected at the AUX3 serial port; 0 flushes the buffer.	1.0
TTY3	66	str	Send length-prefixed string argument out to device connected at AUX2 serial port.	1.0
GETAUX2	67	int	Request to retrieve 1-200 bytes from the device connected at the AUX2 serial port; 0 flushes the buffer.	1.0
CHARGE	68	int	0=release; 1=deploy autocharge-docking mechanism.	1.0
ARM	70-80	int	Pioneer Arm-related commands. See Arm manual for details.	1.1
ROTKP	82	int	Change working rotation Proportional PID value.	1.0
ROTKV	83	int	Change working rotation Derivative PID value.	1.0
ROTKI	84	int	Change working rotation Integral PID value.	1.0
TRANSKP	85	int	Change working translation Proportional PID value.	1.0
TRANSKV	86	int	Change working translation Derivative PID value.	1.0
TRANSKI	87	int	Change working translation Integral PID value.	1.0
REVCOUNT	88	int	Change working differential encoder count.	1.1
DRIFTFACTOR	89	int	Change working drift factor.	1.0
SOUNDTOG	92	int	0=mute User Control piezo; 1 = enable.	1.0
TICKSMM	93	int	Change working encoder ticks per millimeter tire travel.	1.1
BATTEST	250	int	Artificially set the battery voltage; argument in	1.0

			tens volts (100=10V); 0 to revert to real voltage	
RESET	253	none	Force a power on-like reset of the microcontroller.	1.0
MAINTENANCE	255	none	Engage microcontroller maintenance (ARSHstub) mode.	1.0

MOTION COMMANDS AND PARAMETERS

Table 8. Client motion-related commands

ROTATION	
HEAD (#12)	Turn to absolute heading at SETRV max rotational velocity
DHEAD (#13), DCHEAD (#22)	Turn to heading relative to control point at SETRV max velocity
ROTATE (#9)	Rotate at SETRV velocity
RVEL (#21)	Rotate at + (counter) or – (clockwise) deg/sec up to SETRV max
TRANSLATION	
VEL (#11)	Translate forward/reverse at SETV prescribed velocity
MOVE (#8)	Translate forward (+5000 maximum) or backward (-4999mm maximum) distance at SETV velocity
INDEPENDENT WHEEL	
VEL2 (#32)	Set velocity for each side of robot

Note that motion must be enabled using `ENABLE` client command #4 with the argument value of 1. Motion can be disabled with argument value 0.

Client Motion Commands

ARCOS automatically limits `VEL2-`, `VEL-`, and `RVEL`-specified velocities to the parameter values set via `SETVEL` and `SETRV`, flash parameters, and internal platform constants (`TOP` values). Similarly, acceleration and deceleration parameters are limited by internal constants.

The MOVE command (#8) sends the platform forward the specified distance in millimeters if a positive argument, or backward if a negative distance. Each move is limited to +5000 millimeters and -4999 millimeters.

Send an orientation command `HEAD` (#12), `DHEAD` (#13), or `DCHEAD` (#22) with argument value in degrees to turn the robot with respect to its internal dead-reckoned angle to an absolute heading (±0-180 degrees), relative to its immediate heading, or relative to its current heading setpoint (achieved or last commanded heading), respectively. In general, positive relative heading command arguments turn the robot in a counterclockwise direction. However, the robot always turns in the direction that will achieve its heading most efficiently. Accordingly, relative-heading arguments greater than 180 degrees automatically get reduced to 180 or less degrees with a concomitant change in direction of rotation.[9]

The `E_STOP` command #55 or the `STOP` button that is found on some Pioneer robot platforms override deceleration and abruptly stop the robot in the shortest distance and time possible. Accordingly, the robot brakes to zero translation and rotation velocities with very high deceleration and remains stopped until it receives a subsequent translation or

[9] Also when operating the robot with the server-side gyroscope (`hasGyro 2`), the HEAD command gets repeated internally to ensure that the platform reaches its intended heading.

.

rotation velocity command from the client or until the STOP button is reset. (See *E_STOP and E_STALL* later in this chapter.)

DriftFactor, RevCount, and TicksMM Parameters

Three client commands let you change, albeit momentarily for the current client-server connection, those values that affect translation, rotation and drift in your robot. `TicksMM` is the number of encoder ticks per millimeter tire rotation for translation speed and distance computations. The default FLASH value can be changed on-the-fly during a client connection session with the `TICKSMM` client command #93 and unsigned integer value.

`DriftFactor` is a signed value in 1/8192 increments that gets added to or subtracted from the left wheel encoder's ticks to correct for tire circumference differences and consequent translation and rotation drift. `DriftFactor` defaults to its FLASH value on start up or reset, and can be changed on-the-fly with the `DRIFTFACTOR` client command #89 with signed integer argument.

The `RevCount` parameter is the differential number of encoder ticks for a 180-degree rotation of the robot and is used to compute and execute headings. Like `DriftFactor` and `TicksMM`, `RevCount` defaults to its FLASH value on startup or reset, and can be changed on-the-fly with the `REVCOUNT` client command #88 and unsigned integer argument.

SONAR

When connected with and opened by the client, ARCOS automatically begins firing your robot's SONAR, one disc each simultaneously for each of up to four arrays, as initially sequenced and enabled in your robot's FLASH parameters. The SONAR servers also begin sending the SONAR-ranging results to the client via the standard SIP.

Enable/Disabling SONAR

Use the `SONAR` client command #28 to enable or disable all or individual SONAR arrays. Set bit zero of the SONAR argument to one to enable or zero to disable the array(s) SONAR. Set argument bits one through three to an individual array number one through four to enable or disable only that array 1-4. Array zero, the form of the original P2OS command, affects all the arrays at once.

For example, an argument value of one enables all the SONAR arrays, whereas an argument value of six silences array number three. Monitor the status of the SONAR arrays in the `FLAGS` integer bits 1-4 of the standard SIP. The respective bit is set if the array is engaged.

Polling Sequence

Each array's SONAR fire at a rate and in the sequence defined in your robot microcontroller's FLASH parameters. (Consult the next chapter on how to change the FLASH settings.) Use the SONAR `POLLING` command #3 to have your client change the firing sequence, and the `SONAR_CYCLE` command #48 to change the rate. The changes persist until you restart the client-server connection.

The `POLLING` command string argument consists of a sequence of SONAR numbers one through 32. SONAR numbers one through eight get added to the polling sequence for SONAR array number one; numbers nine through 16 get added to the sequence for SONAR array number two; 17-24 specify the sequence for array three; and 25-32 are for array four. You may include up to 16 SONAR numbers in the sequence for any single array. Only those arrays whose SONAR numbers appear in the argument get re-sequenced.

You may repeat a SONAR number two or more times in a sequence. If a SONAR number does not appear in an otherwise altered sequence, the disc will not fire. If you do repeat a SONAR in the sequence, know that ARIA and related clients ignore the first reading if two SONAR ranging data appear in the same SIP.

For compatibility with earlier robot operating systems, if the string is empty, all the SONAR in the array get disabled, but their polling sequences remain unaltered, just as if you had sent the `SONAR` command with an argument value of zero.

Polling Rate

For earlier Pioneer microcontroller versions, the SONAR polling rate was fixed for each array: one SONAR disc per array got polled every 40 milliseconds. So, for an eight-disc array with a sequential polling sequence (12345678 by default), any one SONAR transducer would be read every 320 milliseconds. That common cycle timing accommodates ranging out to the maximum of the SONAR of some five meters for general applications, including features recognition and localization. For other applications, such as close-in obstacle avoidance, a shorter range but faster rate of update is better.

Use the SONAR_CYCLE client command #48 to change the cycle timing on the fly to the command integer's argument value in milliseconds. Minimum and maximum values are two and 120 milliseconds, respectively. The default value is set in FLASH, normally to the legacy 40 milliseconds.

SONAR Range Readings

Only the SONAR readings taken since the last standard SIP get included in the next one. They do not persist beyond that. Readings appear in the standard SIP in three bytes: The first SONAR byte is the SONAR reference, numbered SONAR #0 through #31 (one less than when referenced in client commands). The next two bytes are the little-endian integer representing the range reading in millimeters. ARCOS reports a maximum reading of 5000 when there is no return echo or if the obstacle is closer than about 120 millimeters (~six inches).

STALLS AND EMERGENCIES

With a robot equipped with forward and/or rear bumpers, ARCOS may immediately stop the robot and notify the client of a stall if any one or more of the contact sensors get triggered and the robot is going in the direction of the bump (forward/front or backward/rear). Send the BUMPSTALL command #44 with an integer argument of zero to disable that bump-stall behavior. Give the argument value of one to re-enable BUMPSTALL only when a forward bump sensor gets triggered; two for rear-only BUMPSTALLs; or three for both rear and forward bump contact-activated stalls.

In an emergency, your client may want the robot to stop quickly, not subject to normal deceleration. In that case, send the E_STOP command (#55).

Like BUMPSTALL, use ARCOS' built-in E_STALL feature to simulate a stall when someone presses the robot's STOP button.[10] An integrated switch in the STOP button toggles a dedicated digital I/O port on the microcontroller, thereby notifying ARCOS of the condition. ARCOS stops the robot's motors, puts on the brakes and throws continuous stalls.

Table 9. The FLAGS bits

BIT	CONDITION IF SET
0	Motors enabled
1	SONAR array #1 enabled
2	SONAR array #2 enabled
3	SONAR array #3 enabled
4	SONAR array #4 enabled
5	STOP button pressed
6	E_stall engaged
7	Far ledge detected (IR)
8	Near ledge detected (IR)
9	Joystick button 1 pressed
10	Recharging "power-good"
11-15	Reserved

Unlike other stalls, E_STALL also disables the motors. You must either re-enable the motors manually (MOTORS button) or programmatically (ENABLE command #4).

The E_STALL server notifies your client software through the stall bytes and in bit 5 of the FLAGS byte in the standard so that your client may respond to a STOP E_STALL differently than a regular stall.

Normally enabled (default was disabled in P2OS), change E_STALL by sending the ARCOS command #56. With argument of zero, E_STALL gets disabled. An argument value of one re-enables E_STALL.

ACCESSORY COMMANDS AND PACKETS

Several types of additonal server information packets (SIPs) are also available. On request from the client by a related ARCOS command, the ARCOS server packages and sends one or a continuous stream of information packets to the client over the HOST serial communication line. Extended packets get sent immediately before (such as GYROpac and JOYSTICKpac) or after (such as IOpac) the standard SIP that ARCOS sends to your client every SipCycle milliseconds.[11]

The standard SIP takes priority so you may have to adjust the HOST

[10] Available only on some robots.

[11] You may have to adjust the HOST serial baud rate to accommodate the additional communications traffic.

serial baud rate to accommodate all data packets in the allotted cycle time, or some packets may never get sent.

Packet Processing

Identical with the standard SIP, all ARCOS server information packets get encapsulated with the header (0xFA, 0xFB; 250, 251), byte count, packet type byte and trailing checksum. It is up to the client to parse the packets, sorted by type for content. Please consult the respective client application programming manuals for details.

CONFIGpac and CONFIG Command

Send the `CONFIG` command #18 without an argument to have ARCOS send back a `CONFIGpac` SIP packet type 32 (0x20) containing the robot's operational parameters. Use the `CONFIGpac` to examine many of your robot's default FLASH_based settings and their working values, where appropriate, as changed by other client commands, such as `SETV` and `ROTKV`.

Note that the parameter list may be extended for other Pioneer platforms.

Table 10. CONFIGpac contents

LABEL	DATA	DESCRIPTION
Header	int	Common packet header = 0xFAFB
Byte count	byte	Number of following data bytes
Packet type	byte	CONFIGpac = 0x20
Robot type	str	Typically "Pioneer"
Subtype	str	Identifies the MobileRobots model; e.g. "p3dx-sh",
Sernum	str	Serial number for the robot.
4mots	byte	Antiquated (=1 if AT with P2OS)
Rotveltop	int	Maximum rotation velocity; deg/sec
Transveltop	int	Maximum translation speed; mm/sec
Rotacctop	int	Maximum rotation (de)acceleration; deg/sec^2
Transacctop	int	Maximum translation (de)acceleration; mm/sec^2
PWMmax	int	Maximum motor PWM (limit is 500).
Name	str	Unique name given to your robot.
SIPcycle	byte	Server information packet cycle time; ms.
Hostbaud	byte	Baud rate for client-server HOST serial: 0=9.6k, 1=19.2k, 2=38.4k, 3=56.8k, 4=115.2k.
Auxbaud	byte	Baud rate for AUX1 serial port; see HostBaud.
Gripper	int	0 if no Gripper; else 1.
Front SONAR	int	1 if robot has front SONAR array enabled, else 0.
Rear SONAR	byte	1 if robot has rear SONAR enabled, else 0.
Lowbattery	int	In 1/10 volts; alarm activated when battery charge falls below this value.
Revcount	int	Working number of differential encoder ticks for a 180 degree revolution of the robot.
Watchdog	int	Ms time before robot automatically stops if it has not received a command from the client. Restarts on restoration of connection.

P2mpacs	byte	1 means alternative SIP enable; not used by ARCOS.
Stallval	int	Maximum PWM before stall. If > PWMMAX, never.
Stallcount	int	Ms time after a stall for recovery. Motors lax during this time.
Joyvel	int	Joystick translation velocity setting, mm/sec
Joyrvel	int	Joystick rotation velocity setting in deg/sec
Rotvelmax	int	Current max rotation speed; deg/sec.
Transvelmax	int	Current max translation speed; mm/sec.
Rotacc	int	Current rotation acceleration; deg/ sec^2
Rotdecel	int	Current rotation deceleration; deg/ sec^2
Rotkp	int	Current Proportional PID for rotation
Rotkv	int	Current Derivative PID for rotation
Rotki	int	Current Integral PID for rotation
Transacc	int	Current translation acceleration; mm/ sec^2
Transdecel	int	Current translation deceleration; mm/ sec^2
Transkp	int	Current Proportional PID for translation.
Transkv	int	Current Derivative PID for translation.
Transki	int	Current Integral PID for translation.
Frontbumps	byte	Number of front bumper segments.
Rearbumps	byte	Number of rear bumper segments.
Charger	byte	1 if P3/PeopleBot or 2 if PowerBot automated charger mechanism and circuitry installed in robot; otherwise 0.
SONARcycle	byte	SONAR duty cycle time in milliseconds.
Autobaud	byte	1 if the client can change baud rates; 2 if auto-baud implemented.
HasGyro	byte	1 or 2 if robot equipped with the gyro heading correction device; otherwise 0.
Driftfactor	int	Working drift factor value.
Aux2baud	byte	Baud rate for AUX2 serial port; see HostBaud.
Aux3baud	byte	Baud rate for AUX3 serial port; see HostBaud.
Ticksmm	int	Encoder ticks per millimeter tire motion
Shutdownvolts	int	DC volts X10 at or below which the onboard PC will shut down
		WITH VERSION 1.5...
VersionMajor VersionMinor	str	Null-terminated string for ARCOS version numbers (period char "." separator
GyroCW	int	Gyro calibration factor clockwise
GyroCCW	int	Gyro calibration factor counterclockwise
KinematicsDelay	byte	Time delay (ms) between acquisition and reporting of rotation

NULL	23 bytes	Configuration values not used with Pioneer.

Serial

The baud rates for the HOST and AUX serial ports initially are set from their respective FLASH-based defaults and get reset to those values whenever the microcontroller is reset or upon client disconnection. For advanced serial port management from the client side, ARCOS provides four client commands which let your software reset the HOST (HOSTBAUD #50), Aux1 (AUX1BAUD #51), Aux2 (AUX2BAUD #52) and Aux3 (AUX3BAUD #53) serial port baud rates, respectively. Use the integer command argument values: 0=9600, 1=19.2K, 2=38.8K, 3=57.6K, or 4=115.2K baud, respectively.

With auto-bauding, the HOST serial port automatically reverts to its FLASH default baud rate if, after being reset by the HOSTBAUD client command, it does not receive a subsequent and valid client-command packet within 500 milliseconds.

HOST-to-AUX Serial Transfers

Use the client-side TTY2 command #42 with a string argument to have that string sent out the Aux1 port to the attached serial device, such as a robotic camera. Similarly, use the TTY3 command #66 to send a string argument out the Aux2 port or TTY4 command #60 to send a HOST-mediated client string out the Aux3 port.

ARCOS also maintains three circular buffers for incoming serial data from the respective Aux ports. On request, ARCOS sends successive portions of the buffer to your client via the HOST serial in the respective SERAUXpac (type = 176; 0xB0), SERAUX2pac (type = 184; 0xB8) and SERAUX3 (type = 200; 0xC8) SIPs. Use the GETAUX #43 for Aux1, GETAUX2 command #67 for Aux2 and GETAUX3 command #61 for Aux3.

Use the integer argument value of zero to flush the contents of the respective buffer. Otherwise, use an argument value of up to 253 bytes to have ARCOS wait to collect the requested number of incoming AUX-port serial bytes and them send them in the respective SERAUXpac, SERAUX2pac, or SERAUX3pac SIP.

Encoders

Table 11. ENCODERpac SIP contents

Label	Data	Description
Header	integer	Exactly 0xFA, 0xFB
Byte Count	byte	Number of data bytes + 2 (checksum)
Type	byte	0x90
Left Encoder	integer	Least significant, most significant portion of the
	integer	current accumulated encoder counts from the left wheel
Right Encoder	integer	Least significant, most significant portion of the
	integer	current accumulated encoder counts from the right wheel
Checksum	integer	Checksum for packet integrity

Issue the ENCODER command #19 with an argument of one for a single or with an argument value of two or more for a continuous stream of ENCODERpac (type 144; 0x90) SIPs. Discontinue the packets with the ENCODER command #19 with an argument of zero.

Buzzer Sounds

Pioneer 3 robots have a piezo buzzer on the User Control Panel that aurally notifies you of system conditions, such as low batteries or stalls. For stealthy operation, issue the SOUNTOG command #92 with an argument of zero to mute the microcontroller's buzzer or argument of one to re-enable it. (See also the SOUNDTOG FLASH parameter in the next chapter to set its default state.)

The SAY command #15 lets you play your own sounds through the buzzer. The argument consists of a length-specified string of duration and tone pair bytes. The duration is measured in 20 millisecond increments.

A tone value of zero means silence (musical rest). The next 127 frequencies (1-127) are the corresponding MIDI notes. The remaining tones are frequencies computed as:

Tone – 127 * 32

equivalent frequencies from 1 to 4096, in 32 Hz increments.

Except for the MIDI notes, you'll just have to experiment with tones. Here is the sequence that generates the ARCOS distress wail when the robot stalls or the batteries are low:

```
 50,100,20,0,50,60,0
```

Table 12. System-related buzzer sounds

Condition	Buzzer Sound
Start up	one-time trill
Start client	connection one-time trill
Close client	connection one-time trill
Low battery	repetitive beep
Motor stall	repetitive beep
Joystick calibration†	repetitive beep
Self-test mode†	repetitive beep
Joystick calibrated†	one-time trill
Watchdog timeout	repetitive beep
EStop active†	repetitive beep

† On some robot models

TCM2 COMPASS

The accessory is an integrated inclinometer, magnetometer, thermometer and compass that attaches to one of the Aux serial ports of the ARCOS microcontroller. When attached and enabled (set the TCM2 FLASH parameter to the attached auxiliary serial port number; see next Chapter), special TCM2 compass servers read and report the heading in ±2 degree increments as the compass byte in the standard SIP.

Table 13. TCM2 command #45 arguments

ARCOS Command	Argument (TCM2 Command)	Action
45	0	Module off (software only; not low-power standby)
	1	Compass only (default); reading in standard SIP
	2	Send single TCM2 SIP
	3	Send TCM2 SIPs continuous
	4	Enable user calibration

	5	Enable auto calibration
	6	Stop calibration and send one TCM2 SIP, then revert to default mode 1.
	7	Soft reset (cycle power to hard reset)

Table 14. TCM2 information packet

Label	**Data**	**Value/Description**
Header	integer	0xFAFB
Byte count	byte	23/Number data bytes+checksum
Packet type	byte	0xC0
Pitch	integer	Degrees times ten; maximum pitch depends on TCM2 Module type
Roll	integer	Degrees times ten; maximum roll depends on TCM2 Module type
X	integer	Magnetic field x-component; ±µT times 100
Y	integer	Magnetic field y-component; ±µT times 100
Z	integer	Magnetic field z-component; ±µT times 100
Temperature	integer	Temperature degrees C times 10
Error	integer	Bit-mapped error code (see TCM2 User’s Manual for meanings)
Calibration Scores	byte	H score (0-9)
	byte	V score (0-9)
	integer	M score times 100
Checksum	integer	SIP checksum

Use the `TCM2 client` command #45 to calibrate the device and to request additional information in the form of the `TCM2pac`. ARIA supports the TCM2, as well. Please consult the ARIA documentation for additional information.

Calibration

You should calibrate your TCM2 Module at least once after installing it on your robot. We also recommend that you calibrate it whenever you change operating environments.

To calibrate just the heading, run Aria demo, enter compass mode, engage calibration (user) mode, go into teleoperation mode, and use the left or right arrow key to turn the robot two full revolutions (720 degrees). Then go back to compass mode and stop calibrating. Alternatively, use demo’s ‘d’irect command mode to send the TCM2 client command and argument ’45 4’ to start and ’45 6’ to end user calibrations.

Also, to calibrate for pitch and roll, as well as heading, you need to hold the compass in your hand and turn it side to side while rotating.

The ARCOS TCM2 command #45 mode 4 followed by mode 6 cause the TCM2 module to calibrate and save its settings, then revert back to mode 1 after returning a single TCM2 SIP containing the updated calibration score to the connected client. It and subsequent SIPs contain that latest calibration score on which you may evaluate the calibration settings.

Reset Mode

The reset TCM2 command argument forces a soft Module reset. Cycle AUX power to force a hard reset. When started or upon reset, the TCM2 module reverts to its default state. From the factory, the module is set up to communicate at

9600 baud, include compass, inclinometer, temperature (in degrees Farenheit) and magnetmometer data in its standard output word, be in standby mode, and respond to the single "h" character to halt. Change the module parameters manually with ARCOS-mediated TTY2 commands or through the Windows TCM2 software provided by the manufacturer and included on the TCM2 Compass diskette.

INPUT OUTPUT (I/O)

The robot microcontroller includes several digital and analog I/O ports. Some I/O ports are used by optional accessories. Any ports not used by robot accessories may be used for your own custom I/O

See Appendix A for port locations and specifications.

User I/O

The User I/O connector on the Pioneer 3 microcontroller contains eight digital input (ID0-7) and eight digital output (OD0-7) ports, as well as an analog-to-digital (AN0) port.[12] Additional analog ports are present in the joystick connector and "tilt/roll" connector, and IR and bumper switch input ports may also be used if necessary for custom I/O.

The bit-mapped states of the sixteen digital ports and one of the analog port are sampled, buffered and placed in the standard SIP in the `DIGIN`, `DIGOUT` and `ANALOG` fields. If not physically connected, the digital input and AN port values may vary and change without warning. In addition, if IO Packets are requested, then all digital and analog I/O states as well as IR and bumper states are provided in IO packets.

Send the `DIGOUT` command #30 to set one or more of the eight `DIGOUT` ports on the ARCOS microcontroller. Electrically, the ports are digital high (1) at ~5 VDC (V_{cc}) and low (0) at ~0 VDC (`GND`). `DIGOUT` takes a two-byte (unsigned integer) argument. The first byte is a mask whose bit pattern selects (1) or ignores (0) the state of the corresponding bit in the second byte to set (1) or unset (0) the digital output port.

For example, here's the ARCOS client command to set digital output ports zero and three (`OD0` and `OD3`), reset port four (`OD4`), and leave all the rest alone:

```
 250, 251, 6, 30, 27, 25, 9, 55, 36
```

See Appendix A for details on connecting devices to User IO including port locations and specifications.

IO Packets

Client software may request the IO Packet ("`IOpac`") SIP. This packet has type 240 (0xF0). It provides the current state of all robot I/O.

Send the `IOREQUEST` command number 40 with an argument value of 0, 1 or 2 to request IO packets. The argument value 1 requests a single packet to be sent by the next client-server communications cycle. The request argument value of 2 tells ARCOS to send `IOpac` packets continuously, at approximately one per cycle depending on serial port speed and other pending SIPs. Use the `IOREQUEST` argument value 0 to stop continuous `IOpac` packets.

Table 15. IOpac packet contents

LABEL	DATA	VALUE	DESCRIPTION
Header	2	0xFA, 0xFB	Common header
Byte count	1	22	Number of data bytes + 2
Type	1	0xF0	Packet type
N DIGIN	1	4	Number of digital input bytes
DIGIN	1	varies 0-255	ID0-8 bits mapped
Frontbumps*	1	varies 0-255	Front bumper bits mapped

[12] Many of these ports are used by the Gripper and other accessories. Alternative I/O is available.

Rearbumps*	1	varies 0-255	Rear bumper bits mapped
IRs	1	varies 0-255	IR inputs
N DIGOUT	1	1	Number of digital output bytes
DIGOUT	1	varies 0-255	Digital output byte(s)
N AN	1	9	Number of A/D values
AN	10	5 integers varying 0-1023	Analog ports 0-7 input values at 10-bit resolution: 0-1023 = 0-5 VDC
Battery	2	0-1023	Battery analog input (AN3 Pioneer 3)
Checksum	2	varies	Computed checksum

* Actual, not affected by `InvertBumps` since bumper bits may be used for other digital input besides bumpers.

Bumper and IR I/O

Two 10-position microfit connectors on the ARCOS microcontroller provide 16 digital input ports that are normally used for the bumper accessory, but also available for your own attachments. See *Appendix A* for connector details.

Similarly, the Motor-Power connector on the microcontroller contains eight digital inputs that we normally use for IR sensors on the Performance PeopleBot and PowerBot, and whose states are digitally mapped. See *Appendix B* for connector details.

Normally pulled high (5 VDC=digital port bit value 1), all the bumper and IR bit-mapped switches go low (digital 0) when the respective port gets triggered. Bumper inputs also appear with the stall bits in the standard SIP, but unlike in the `IOpac`, are modified by the `InvertBumps` mask. All the bumper and IR data bits appear in the `IOpac` packet.

Additionally, the IR inputs may be mapped in the FLASH parameters to corresponding bumper inputs for reflexive-like reactions, such as bump-related stalls. See *Chapter 7, Updating and Reconfiguring ARCOS*, for details.

JOYSTICK

Use the ARCOS client `JOYREQUEST` command number 17 with an argument value of zero, one or two to request information about the joystick, if it is enabled (see next Chapter). The argument value one requests a single packet (type = 248; 0xF8) to be sent by the next client-server communications cycle. The request argument value of two tells ARCOS to send `JOYSTICKpac` packets continuously, at approximately one per cycle depending on serial port speed and other pending SIPs. Use the `JOYREQUEST` argument value zero to stop continuous `JOYSTICKpac` packets.

LABEL	**DATA**	**VALUE**	**DESCRIPTION**
header	2	0xFA, 0xFB	Common header
Byte count	1	11	Varies
type	1	0xF8	Packet type
button0	1	0 or 1	1=button pressed
button1	1	0 or 1	1=button pressed
X-axis	2	varies 0-1023	Rotation analog
Y-axis	2	varies 0-1023	Translation analog
throttle	2	varies 0-1023	Throttle setting
checksum	2	varies	Computed checksum

GRIPPER

Please consult the respective Gripper manual for details.

ARCOS supports a `GRIPPERpac` (type=224; 0xE0) packet type and related `GRIPREQUEST` command #37 to retrieve setup and status information from the servers.

Normally disabled, your client program may request one or a continuous stream (command argument greater than one) of Gripper packets. Send `GRIPREQUEST` with the argument value zero to stop continuous packets.

Table 16.

LABEL	DATA	DESCRIPTION
header	int	Exactly 0xFA, 0xFB
byte count	byte	Number of data bytes + 2 (checksum)
type	byte	Packet type = 0xE0
hasgripper	byte	Gripper type: 0=none; 1=Pioneer; 2=PeopleBot
grip_state	byte	*See* Table 16 below.
grasp_time	byte	Ms time controls grasping pressure.
checksum	integer	Computed checksum

GRIPPERpac GRIP_STATE byte

	BIT	MEANING WHEN SET (1)
G	0	Paddles open[1]
R	1	Paddles close[1]
I	2	Paddles moving
P	3	Gripper error
L	4	Lift up[1]
I	5	✓ Lift down[1]
F	6	Lift moving
T	7	Lift error

Note that the Gripper status information bits may also may be obtained from the respective `DIGIN` and `DIGOUT` values of the standard SIP as related to the User I/O port states. See *Appendix A* for connection details.

HEADING CORRECTION GYRO

With the gyroscope accessory, your client software (ARIA 1.3 and later) or ARCOS itself (ARCOS 2.0 and later) may detect and compensate for robot heading changes that aren't detected by the encoders, such as from slipping wheels. The microcontroller supports the gyro via attachment to its AN6 and AN7 analog-to-digital input ports.

To enable the gyro, you must set the `HasGyro` FLASH parameter to either value one for client- or two for server-side management using the ARCOScf tool (see next chapter). Set it to zero if the gyro isn't attached or you want to disable the gyro.

ARCOS collects and averages 10-bit (0-1023) gyro rate and 8-bit (0-255) temperature data over a 25 millisecond time period—the bandwidth of the gyro. When not moving, the gyro rate is centered around 512 or so, depending on the gyro's temperature and other calibration factors which drift with use and should be corrected on the fly. Values below that center point indicate counterclockwise rotation rates; values above the resting center measure clockwise rotation rates. The gyro rate maximum is 300 degrees per second.

What happens next depends on the `HasGyro` setting.

Client-Side Gyro

With `HasGyro` set to one (legacy), ARCOS simply collects the gyro readings and will, upon request by the client with command GYROREQUEST #58 and argument 1 (zero to cancel), send the collected data just before the standard SIP to a connected client in a `GYROpac` (type=0x98) server information packet for processing. Analysis of the gyro data and subsequent modifications to the robot's heading are done on the client side, as supported in the latest versions (1.3 and later) of ARIA. See `Aria/src/ArAnalogGyro.cpp` for details. Note that the client-side gyro calibration/correction factor is found as the `GyroScaler` value in your robot's Aria/params/*.p parameter file, and should be calibrated for the particular robot and gyro. See Chapter 7, Calibration & Maintenance for details.

`GYROpac` consists of a count byte of the rate and temperature data pairs accumulated since the last cycle (typically four for a 100 millisecond cycle time), followed by that number of rate/temperature integer/byte pairs.

Table 17. GYROpac SIP contents

LABEL	DATA	VALUE	DESCRIPTION
Header	2	0xFA, 0xFB	Common header
Byte count	1	xx	Varies
Type	1	0x98	Packet type
N pairs	1	x	Number of gyro data pairs
FOR N PAIRS: …			
Rate	2	varies 0-1023	Gyro rate
Temperature	1	varies 0-255	Gyro temperature
Checksum	2	varies	Computed checksum

Server-Side Gyro

With the FLASH parameter `HasGyro` set to two, ARCOS manages the gyro internally. It, too, averages readings over the 25 millisecond time, but it does not send the GYROpac. Instead, ARCOS fuses the gyro and encoder readings to compute the XPos, YPos, and ThPos in the standard SIP.[13]

Like ARIA's treatment, the server-side gyro firmware automatically detects its center point and compensates for drift. Unlike the client side, the server-integrated gyro has two FLASH-based calibration factors—`GyroCW` and `GyroCCW`--that determine actual rates clockwise and counterclockwise and need to be set for each robot/gyro pair—something typically done at the factory, but should be recalibrated periodically. Client commands #38 (GYROCALCW) and #39 (GYROCALCCW) let you temporarily change the clockwise and counterclockwise values, respectively, to help during calibration. See Chapter 7, Calibration & Maintenance for details.

If the server-side gyro is malfunctioning, such as if it's unable or has yet to determine its center point, the robot will stall, complete with excessive beeping from the piezo buzzer. The GYROREQUEST command #58 with argument zero disables the client-side gyro; an argument of one re-enables it.

[13] With no gyro or with the client-side gyro, position integration is based on encoder readings exclusively.

Chapter 6 Updating & Reconfiguring ARCOS

The ARCOS firmware and a set of operating parameters get stored in your Pioneer 3 microcontroller's FLASH memory. With special upload and configuration software tools, you change and update ARCOS, too. No hardware modification is required.

Where to Get ARCOS Software

Your Pioneer 3 robot comes preinstalled with the latest version of ARCOS. And the various ARCOS configuration and update tools come with the robot on CD-ROM. Thereafter, stay tuned to the `pioneer-users` newsgroup or periodically visit our support website to obtain the latest ARCOS software and related documentation:

`http://robots.MobileRobots.com`

The main utility, `ARCOScf`, is a multi-functional application for both uploading new ARCOS versions as well as modifying your robot's onboard FLASH-based parameters.

ARCOS Maintenance Mode

To connect with and update your robot's ARCOS servers and its FLASH-based operating parameters, you need to first connect a serial port on the PC from which you will run ARCOScf to the HOST port of your robot's microcontroller:

- ✓ If you are running from an onboard PC, the computer-to-`HOST` connection already is made.
- ✓ If you have an onboard PC, but prefer to use an external computer for maintenance, simply power down the onboard computer.
- ✓ If you use radio or Ethernet wireless, switch its power OFF (typically AUX1).
- ✓ When connecting from an external PC, directly tether (no radios) its serial port to the 9-pin DSUB SERIAL connector on the User Control Panel.

Enabling Maintenance Mode

When in maintenance mode, the robot can only be configured or reflashed using ARCOScf. (When in normal server mode, any software can connect and control all the features of the robot).

Use maintenance mode when there is a problem with the ARCOS firmware that prevents startup in normal mode.

To enter maintenance mode, hold down the white MOTORS button, then press the red RESET button and release. Press and release the red RESET button to re-enter normal mode.

If for any reason your robot's FLASH parameters get erased or your ARCOS firmware encounters a code fault, your Pioneer 3 microcontroller automatically reverts to its ARSHstub-based maintenance mode.[14] Or if you attach a PC to the SERIAL port on the User Control Panel, open the port through software such as by running ARCOScf on that PC, and then reset or otherwise restart the microcontroller, it will automatically revert into maintenance mode.

Like with previous Pioneer microcontrollers, you may manually engage maintenance code:

1. Press and hold the white `MOTORS` button on the User Control Panel
2. Press and release the adjacent red `RESET` button
3. Release the `MOTORS` button.

Unlike previous microcontrollers and certainly with much more convenience since you don't need to be fiddling with buttons, ARCOScf *automatically* engages maintenance mode. Just start ARCOScf and it forces the microcontroller into maintenance mode. It uses the ARCOS command #255 to do that.

The `STATUS` LED on the User Control Panel should flash twice the rate than when in server ("wait") mode and the `BATTERY` LED should shine bright red.

[14] ARSHstub is a resident GDB-like interface for FLASH updates and other debugging uses.

ARCOScf

The ARCOS update and configuration program, `ARCOScf`, is part of a collection of utilities and files for comprehensive management of your Pioneer 3 robot's onboard servers and FLASH-based operating parameters. The distribution archive for the software is simply named `ARCOSV_v` (`V` and `v` are the version major and minor numbers, such as 1_0), with a "`.tgz`" suffix for Linux-based PCs or "`.exe`" for Windows computers.

Install the utilities and files on the PC you plan to use for maintaining your robot's operating system and parameters by double-clicking the distribution software's onscreen icon or otherwise executing the self-extracting, self-installing package. For Linux, `uncompress` and `untar` the files. For example,

```
% tar -zxvf ARCOS1_0.tgz
```

The expanded archive creates an `ARCOS/` directory in the selected Windows or current Linux path and stores the ARCOS software within.

STARTING ARCOSCF

`ARCOScf` is a text-based console application which like `demo` is built with ARIA. To start it from a console terminal under Linux, for example, navigate to the ARCOS directory and invoke the program:

```
 % cd /usr/local/ARCOS
 % ./ARCOScf <options>
```

With Windows PCs, you may double-click the `ARCOScf` icon to automatically open a console window and start the program without any options. To start up with command-line options, `Run` the program from the `Start` menu, or run `Command` from the `Start` menu, then navigate to the ARCOS directory and start `ARCOScf` with options.

For example (after invoking the MSDOS-like command window):

```
 C:\> cd Program Files\MobileRobots\ARCOS
 C:\Program Files\MobileRobots\ARCOS\> ARCOScf <options>
```

Normally (without any command-line arguments), `ARCOScf` starts up expecting to connect with ARCOS through the PC's `COM1` or `/dev/ttyS0` serial port. If successfully connected, the program automatically retrieves your robot's FLASH-stored operating parameters and enters interactive mode.

Start Up Arguments

`ARCOScf` runs in two stages: startup followed by interactive mode. When invoked, you may start `ARCOScf` with various ARIA and other command-line options. Preface each option with a dash ("-"), followed by the argument and argument value, separated by spaces. For example

```
   ARCOScf -n
```

starts up the program without connecting with the microcontroller, so that you may examine and maintain disk-based copies of your robot's operating parameters.

Table 18. ARCOS start-up options

ARGUMENT	VALUE	DESCRIPTION
-b	batch commands	Batch mode executes list of ARCOScf interactive-like mode commands with arguments, then exits automatically.
-u	motfile	Automatically upload an ARCOS (.mot) motfile after connecting with the microcontroller.
-l	paramsfile	Load the disk-stored params (.rop) file instead of the robot's copy
-n	none	Don't connect with the microcontroller
-rp	serial-device	Uses specified serial port for connection
-rb	baudrate	Specify the port connection baud rate

-s	paramsfile	On exit from ARCOScf, automatically save the current parameter values to the named .rop paramsfile

To start up ARCOScf and make a connection with a serial port other than the default COM1 or ttyS0:

```
C:\Program Files\MobileRobots\ARCOS> ARCOScf -rp COM3
```

Similarly, these startup arguments tells ARCOScf to upload a fresh copy of the firmware to your robot's microcontroller and then exits:

```
% ./ARCOScf -u ARCOS1_0.mot -n -b
```

CONFIGURING ARCOS PARAMETERS

Your robot has several parameters stored in FLASH that ARCOS uses to configure its servers and auxiliary attachments and to uniquely identify your robot. For instance, the default maximum translation velocity is stored in the TransVelMax parameter. Its value takes effect when starting your robot or after resetting the microcontroller, and may be changed temporarily by a client command. Use ARCOScf's batch or interactive modes to modify these operating parameters, and hence your robot's default operating characteristics.

Start up ARCOScf as described in the previous section. As discussed earlier, ARCOScf normally downloads the set of operating parameters from your robot's FLASH for your review and modification. Or you may load a disk-stored version of those parameters.

Interactive Commands

To operate ARCOScf in interactive mode, simply type a keyword at its command prompt. Some keywords affect the operation of ARCOScf or the status of the parameters file as a whole. For instance, to review the list of current ARCOS FLASH variables, type 'v' or 'view' followed by a return (Enter). Each successive return will display additional variable keywords and current values. Similarly, type '?' or 'help' to see a list of ARCOScf interactive commands.

Changing Parameters

Most keywords refer to the operating parameters themselves. Alone, a parameter's keyword simply asks ARCOScf to display the parameter's value. Provide an value with the parameter keyword separated by a space to change it. That value may be a string (no quotes or spaces) or a decimal or hexadecimal ("0xN") number. For example, to change the watchdog timeout to four seconds, type:

```
> watchdog 4000
```

or

```
> watchdog 0xfa0
```

See the respective control command and parameter Tables nearby for a full description of ARCOScf operation.

SAVE YOUR WORK

While changing parameter values in ARCOScf interactive mode, you are editing a temporary copy; your changes are not put into effect in your robot's FLASH until you explicitly "save" them to the microcontroller.

Also use the ARCOScf save command to save a copy of the parameters to a disk file for later upload. We strongly recommend that you save each version of your robot's parameter values to disk for later retrieval should your microcontroller get damaged or its FLASH inadvertently erased. Default parameter files come with each ARCOS distribution, but it is tedious to reconstruct an individual robot's unique configuration.

Table 19. ARCOScf interactive commands

COMMAND	DESCRIPTION

keyword <value>	Alone, a keyword displays current, edited value. Include a value to change it.
v or view	Display FLASH parameters.
u or upload <motfile>	Upload specified ARCOS image (.mot) file to the microcontroller.
r or restore <paramsfile>	Restore parameters to values currently stored in FLASH or, if given, from a paramsfile (.rop) on disk
save <paramsfile>	Saves current edited values to FLASH or saves current edited values to paramsfile (.rop) on disk for later reference.
q or quit	Exits ARCOScf.
? or help	Displays these commands and descriptions.

Table 20. Sample ARCOS FLASH configuration parameters with values for Pioneer 3-DX

KEYWORD	DATA	DEFAULT	DESCRIPTION
type	str	Pioneer	Identifies the robot type.
subtype	str	P3DX-SH	Identifies the robot model.
name	str	P3-DX	Unique name for your robot. Maximum of 20 characters, no spaces.
sernum	str	Serial	Serial number for the robot.
ticksmm	int	128	Encoder ticks/mm: (4 x ticks per rev x gear-ratio)/ (wheel_diameter x Π)
revcount	int	16570	The number of differential encoder ticks for a 180-degree revolution of the robot.
driftfactor	int	0	Value in 1/8192 increments to be added or subtracted from the left encoder ticks in order to compensate for tire differences.
battconv	byte	0	0 if a 12V system; 1 if 24V
lowbattery	int	115	In 1/10 volts; microcontroller alarm activated when battery charge falls below this value.
shutdownvolts	int	108	In 1/10 volts; microcontroller disconnects client and signals onboard PC to shutdown when battery charge falls below this value.
hostbaud	byte	0	Baud rate for client-server HOST serial: 0=9.6k, 1=19.2k, 2=38.4k, 3=56.8k, 4=115.2k.
auxbaud1	byte	0	Baud rate for AUX serial port 1; see HostBaud
auxbaud2	byte	0	Baud rate for AUX serial port 2; see HostBaud
auxbaud3	byte	0	Baud rate for AUX serial port 3; see HostBaud
sipcycle	byte	100	Server information packet cycle time in 1 ms increments. Default is classic 100 ms.
watchdog	int	2000	Ms time before robot automatically stops if it has not received a command from a client. Restarts on

			restoration of connection.
soundtog	byte	1	0 disables the buzzer
SONARcycle	byte	40	SONAR cycle time in milliseconds
SONAR1	str	12345678	Ping sequence for SONAR array #1. Up to 16 number characters 1-8; 0 to disable the array
SONAR2	str	0	Ping sequence for array #2. See SONAR1 above
SONAR3	str	0	Ping sequence for array #3. See SONAR1 above
SONAR4	str	0	Ping sequence for array #4. See SONAR1 above
hasgyro	byte	0	Set to 1 (client-side) or 2 (server-side) if you have the gyro accessory
gyrocw	int	1000	Adjust for gyro type 2 clockwise rotations
gyroccw	int	1000	Adjust for gyro type 2 counter-clockwise rotations
gyroratelimit	int	0	Degs per sec X 10 below which to not believe the gyro; 0 = internal default of 20 = 2 deg/sec
gyrorange	int	0	Max rotational speed of gyro (SPI-based only); 0 internal default = 320 deg/sec
hasbrakes	byte	0	1 if your robot has brakes; 0 if not.
charger	byte	0	Set to 1 if P3/PeopleBot or 2 if PowerBot autocharger mechanism and circuitry installed; otherwise 0
chargethreshold	int	0	PowerBot only
gripper	byte	0	Set to 1 if DX/AT Gripper; 2 if Gripper on Performance PeopleBot
tcm2	byte	0	TCM2 module, if connected, specify AUX port number 1, 2, or 3
lcd	byte	0	LCD module, if attached, specify AUX port number 1, 2, or 3
uichange	byte	1	0 restricts number of user options selectable with LCD
frontbumps	byte	0	Number of front bumper segments
rearbumps	byte	0	Number of rear bumper segments
invertbumps	byte	0	0=none; 1=front; 2=rear; or 3=invert both; affects STALL bits in std. SIP only.
bumpstall	byte	0	3=disable bump stall; 1=enable rear; 2=enable front; 0=enable both front and rear bump stalls
frontirs	str	0	Associates IR ports with respective front bumpers
rearirs	str	0	Associates IR ports with respective rear bumpers
invertirs	byte	0	0=none; 1=front; 2=rear; or 3=invert both; only related to bumper responses.
stallval	int	200	Maximum PWM before stall. If > PwmMax, never.
stallcount	int	200	Ms time motors disabled after a stall for recovery.
pwmmax	int	500	Maximum motor PWM (500 maximum).
rotveltop	int	360	Maximum rotation velocity; deg/sec

`transveltop`	`int`	`1500`	`Maximum translation speed; mm/sec`
`rotacctop`	`int`	`300`	`Maximum rotation (de)acceleration; deg/sec`^{2}
`transacctop`	`int`	`2000`	`Maximum translation (de)acceleration; mm/sec`^{2}
`rotvelmax`	`int`	`100`	`Max rotation speed; deg/sec.`
`transvelmax`	`int`	`750`	`Max translation speed; mm/sec.`
`rotacc`	`int`	`100`	`Rotation acceleration; deg/sec`^{2}
`rotdecel`	`int`	`100`	`Rotation deceleration; deg/sec`^{2}
`rotkp`	`int`	`40`	`Proportional PID for rotation`
`rotkv`	`int`	`20`	`Differential PID for rotation`
`rotki`	`int`	`0`	`Integral PID for rotation`
`transacc`	`int`	`300`	`Translation acceleration; mm/sec`^{2}
`transdecel`	`int`	`300`	`Translation deceleration; mm/sec`^{2}
`transkp`	`int`	`40`	`Proportional PID for translation`
`transkv`	`int`	`30`	`Differential PID for translation`
`transki`	`int`	`0`	`Integral PID for translation`
`joystick`	`byte`	`0`	`Joystick type: 0=analog, 1=inductive`
`joysafe`	`byte`	`0`	`1 lets you operate in joydrive UNSAFE mode`
`joyvelmax`	`int`	`750`	`Joydrive maximum translation velocity`
`joyrvelmax`	`int`	`50`	`Joydrive maximum rotation  velocity`

PID Parameters

The ARCOS configuration parameters include settings for the PID motor controls for translation and rotation of the robot. The translation values also are used for independent-wheel mode. The default values are for a lightly loaded robot. Experiment with different values to improve the performance of your robot in its current environment.

The Proportional PID (Kp) values control the responsiveness of your robot. Lower values make for a slower system; higher values make the robot "zippier", but can lead to overshoot and oscillation.

The Derivative PID (Kv) dampens oscillation and overshoot. Increasing values gives better control of oscillation and overshoot, but they also make the robot's movements more sluggish.

The Integral PID (Ki) adjusts residual error in turning and velocity. Higher values make the robot correct increasingly smaller errors between its desired and actual angular position and speed.

DriftFactor, TicksMM and RevCount

ARCOS uses the `TicksMM` and `RevCount` parameters to convert your platform-independent speed and rotation commands—typically expressed in millimeters or degrees, respectively—into platform-dependent units. And it uses `DriftFactor` to compensate for tire differences.

The `TicksMM` value is the number of encoder pulses ("ticks") per millimeter of wheel rotation. The value is, of course, dependent upon the wheel encoder's resolution, the motor-to-wheel gear ratio and the wheel's diameter.

The `RevCount` value is the differential number of encoder ticks for a 180-degree turn of the robot. It depends on a number of factors, principally the length of the wheel base which may change due to payload, tire wear, operating surface and so on.

The DriftFactor is a value in 1/8192 units that gets added or subtracted from the left-wheel encoder count at each motor cycle. In doing so, it compensates for tire difference and thereby straightens the robot's translation forward and backward.

Table 21. Some platform-dependent parameters

PARAMETER	VALUES		
	P3DX	P3AT	PERF PB
encoder ticks/rev	500	500	500
gear ratio	38.3	49.8	38.3
wheel diam (mm)	195	220	195
encoder ticks/mm	132	138	132

The TicksMM and RevCount parameters affect the conversion of your motion command arguments into platform-dependent values used by ARCOS. Unlike previous firmware implementations, ARCOS also uses TicksMM and RevCount to convert its internal measures into platform-independent position and velocity values that it sends to the client in the standard SIP, such as Xpos and Ypos. Accordingly, you'll notice that the respective ARIA client parameters have many conversion factors like DistConvFactor set to 1.0.

STALLVAL AND STALLCOUNT

An ARCOS stall monitor maintains a running average of PWM values for each wheel over a 500 millisecond integration period. PWM values get added to the sum if the wheel speed is below 100 mm/sec. The average is then compared with the stallVal FLASH value. If it exceeds that value, in other words the motors are being given lots of power but are barely moving if at all, a stall occurs. Once stalled, power is removed and the motors relax for the stallWait period, after which power gets reapplied.

BUMPERS

Use the bumpStall FLASH parameter to set the default for the robot's behavior when its front and/or rear bumper gets triggered. Normally, bumpStall is engaged for both front and rear (default value of 0) bumpers. Reset it to 3 to disengage bump-related stalls altogether; 1 to trigger stalls only when the rear bumpers engage; or 2 for front bumps only.

You may over-ride the bumpStall FLASH default with the BUMPSTALL client command #44, although the command arguments are the reverse: enabling versus disabling the various bumper-stall combinations. Your robot's bumpStall behavior reverts to the FLASH default on reset and up disconnection from the client.

ARCOS implements three FLASH parameters that specify states and numbers of front and rear bumper segments. Set the frontBumps and rearBumps values to the number of bumper segments for the front and rear bumpers, respectively; or to 0 if you don't have a particular bumper. The number of segments is used to isolate the bumper bits, if any, so that a triggered bumper event is reported correctly in the STALL values of the standard SIP. Use the invertBump FLASH parameter to invert those bumper-related STALL values, but not the hardware-related states reported in the IOpac.

The frontBumps and rearBumps byte values also are reported near the end of the CONFIGpac. If for any reason you remove a bumper from your robot, you MUST reset the associated frontBumps or rearBumps FLASH value. Otherwise, the robot will stall incessantly and ARIA won't let you drive.

IRS

Use the frontIRs and rearIRs FLASH parameters to associate the IR/extended bumper ports with a front and/or rear bumper, respectively. Each parameter value is a string of eight characters. Each character, in sequence, represents the

IR ports 1-8. Set the value of each character position to the number of the bumper you want that IR port to effect; otherwise, '0' for no association/IR bumper.

For example, the `frontIRs` string "002411" means that IR #3, when triggered, will cause a bump event as if the front bumper #2 had been hit. Similarly, IRs 5 and 6 will cause bump events on the front bumper #1; redundancy is okay. We aren't using IRs #1 and #2, at least not with the front bumpers. Note, too, that there are only six characters in the string, even though there are a total of eight IR ports. Trailing IR spaces automatically get set to zero.

The IR inputs are pulled up electrically, and so their digital states normally are high ("1") when parsed in the IO packet and go low ("0") when shorted to ground. Use `invertIRs` to invert that logic so that the related `frontIRs` and `rearIRs` effects are inverted.

README

We often update ARCOS with new and experimental features. Be sure to read the text README file found with each distribution for more FLASH parameter details and operations.

Chapter 7 Calibration & Maintenance

Your Pioneer 3 robot platform is built to last a lifetime and requires little maintenance.

TIRE INFLATION

Maintain even tire inflation for proper navigation of your Pioneer 3-AT. We ship with each pneumatic tire inflated to 23 psi. If you change the inflation, remember to adjust the `driftFactor`, `ticksMM`, and `revCount` FLASH values.

CALIBRATING YOUR ROBOT

Your robot comes with FLASH parameters adjusted for operation on smooth, flat surfaces with its original payload. If you operate your robot on some different surface and with lighter or heavier loads, you probably will need to recalibrate many of its operating parameters, such as `driftFactor`, `revCount`, `ticksMM` and the PIDs.

The ARIA `demo` program has two modes to help you do that. In '`p`'osition mode, `demo` displays current heading and position. Press the '`r`' key to reset these to 0 at any time. Press the right or left arrow key to have the robot rotate 90 degrees either clockwise or counterclockwise, respectively. The up and down arrow keys tell the robot to advance forward or backward one meter, respectively.

ARIA `demo`'s '`d`'irect mode lets you change the variety of operating parameters on-the-fly by letting you send an ARCOS client command number and value to be used during the current session. To replace the default values in FLASH, use `ARCOScf`.

Accordingly, to properly calibrate your robot, first use `ARCOScf` and record and save on disk the current values for the respective parameters, such as for `driftFactor` and `revCount`. Then connect with the ARIA `demo` and engage position mode to move the robot. As accurately as possible—a laser-line or pointer is helpful here--measure its actual motion and position and use `demo`'s direct mode to adjust the reported values for your robot's current configuration and operating environment.

Note:

Disable the server-side gyroscope accessory (HasGyro 2) before calibrating driftFactor and revCount.

Standard Calibrations

Start with `driftFactor` since its value affects both `ticksMM` and `revCount`. Draw a line on the floor parallel to the robot's translation travel and drive the robot forward or back at least three meters. Adjust `driftFactor` (command #89) to minimize the robot's drift off that line.

Then drive the robot forward or back one or more meters and compare its actual translation distance you accurately measure with `demo`'s ARCOS-reported distance (x) in millimeters. Adjust `ticksMM` (command #93) so that the numbers match.

Likewise, rotate your robot and compare your measured rotation angle to the reported heading (`th`). Adjust `revCount`-- the measure of differential encoder ticks to achieve 180-degrees rotation-- accordingly (command #88).

Finally, drive the robot around and adjust its PID, velocity and acceleration values to achieve the desired performance for the operating configuration.

Gyroscope Calibrations

With the client-side gyroscope (`hasGyro` set to one), ARIA retrieves the rate data and manages it from the client side. Consequently, you don't have to disable it when performing calibrations with the ARIA demo program. In fact demo's 'p'osition mode displays the gyro versus encoder headings separately for tandem calibration. Adjust the `GyroScaler` value in your robot's `Aria/params/*.p` params file (`p3dx-sh.p`, for example).

When using the server-side mode for the gyro, disable it when doing the standard encoder calibrations, such as with the client command #58 and argument 0 or by simply setting `HasGyro` to 0. Then, after re-enabling it, adjust the `GyroCW` and `GyroCCW` values as you did for `revCount`. Use the `GYROCALCW` (#38) and `GYROCALCCW` (#39) client commands to refine the values before committing them to FLASH.

When you are satisfied that the robot moves and rotates the proper distances and headings, and drives with the proper performance, commit those new values into their related FLASH parameters in your robot with `ARCOScf` and don't forget to save a copy in a `.rop` file for later reference.

DRIVE LUBRICATION

Pioneer 3 drive motors and gearboxes are sealed and self-lubricating, so you need not fuss with grease or oil. An occasional drop or two of oil on the axle bushings between the wheels and the case won't hurt. And keep the axles clear of carpet or other strings that may wrap around and bind up your robot's drive.

BATTERIES

Lead-acid batteries like those in your Pioneer 3 robot last longest when kept fully charged. In fact, severe discharge is harmful to the battery, so be careful not to operate the robot if the battery voltage falls below 11 VDC. You may charge the batteries as often as needed or as long as needed, this will not damage the batteries or change their behavior.

Changing Batteries

CAREFUL!
The Batteries slide in
TERMINALS LAST!

To access the batteries, unlatch the rear door, swing it open and locate the one to three onboard batteries inside.

To remove a battery, simply grasp it and pull out. We provide a suction-cup tool to help.

Insert the batteries by simply sliding each one into a battery box compartment. Load the batteries so that their weight gets distributed evenly across the platform: If using a single battery, place it in the center. If using two batteries, place one at either side. If using three batteries, use all three locations.

Make sure that all battery terminals are making contact with the spring contacts in the robot. If any batteries fail to make full contact, then damage may result to the batteries or robot.

CAREFUL
Always ensure that battery terminals are making full contact with the robot's spring contacts.

Hot-Swapping the Batteries

You may change the batteries on your Pioneer 3 without disrupting operation of the onboard systems (except the motors, of course): Either connect the charger, which powers the robot's systems while you change the battery or batteries. Or, if you have two or three batteries, swap each with a freshly charged one individually, so that at least one battery is in place and providing the necessary power.

Charging the Batteries

If you have the standard charger accessory, insert it into a standard 110 or 220 (Europe/South America/Asia) VAC wall power receptacle. (Some users may require a special power adapter.) Locate the round plug at the end of the cable that is attached to the charger and insert it into the charge socket that is just below your robot's Main Power switch. The LEDs on the charger indicate charge status, as marked on its case.

It takes fewer than 12 hours—often just a few hours, depending on the level of discharge—to fully charge a battery using the accompanying charger (roughly, three hours per volt per battery). Although you may operate the robot while recharging, it restricts the robot's mobility.

Automated Docking/Charging System

The automated docking/charging system accessory optimally conditions power to charge the three 21-Ahr, 12 VDC lead-acid batteries (6 A charging current max) and provides sufficient power (up to 5A) for operation of all onboard systems.

The charging mechanism and onboard power conditioning circuitry can be retrofitted to all Pioneer 3 and some Pioneer 2 and PeopleBot robots. All require return to the factory.

Alternative Battery Chargers

The center post of the charger socket is the positive (+) side of the battery; the case is the negative (-) side. A diode protects against the wrong charger polarity. Nonetheless, if you choose to use an alternative battery charger, be sure to connect positive to positive and negative to negative from charger to robot.

An alternative AC to DC converter/battery charger should sustain at least 0.75A at 13.75 to 14 VDC per battery, and not more than 2-2.5 amperes per battery. The High-Speed Charger accessory, for example, is a four ampere charger and should be used with at least two of the standard batteries.

An alternative charger also should be voltage-and current-limited so that it cannot overcharge the batteries.

TIGHTENING THE AT DRIVE BELT

Occasionally, particularly after heavy use, the Pioneer 3- or 2-AT drive belts that mechanically link the front and rear motors on each side will loosen and slip, resulting in a load popping noise. To tighten them, start with a 3mm hex key to loosen, but not remove, the three screws on the side of the robot near the front wheel. One screw is partly behind the wheel, so with our parts kit, we included a 3mm hex key with a shortened "L" section to fit behind the wheel.

Figure 13. Locations of the P3-AT's belt-tensioning bolts

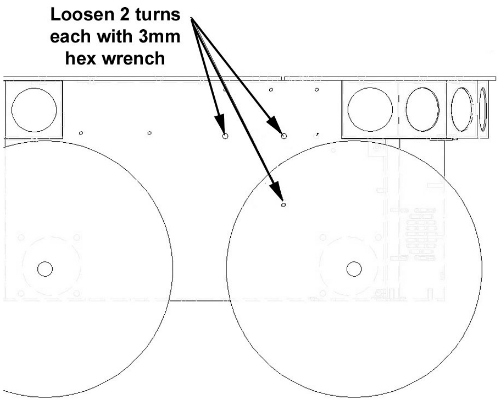


Figure 14. Loosen the AT drive belt retainer screws first.

Remove the small plastic plug which is near the hinge on the top plate and near the edge by the wheel. Under it, you will see the head of a large hex bolt. This bolt tightens (clockwise) or loosens (counter-clockwise) the drive belt for that side of the robot. Turn it using a 5mm hex key probably not more than 1 full rotation. Avoid over tightening.

Test to make sure that it is tight enough by holding the wheel while running the self test. When adjusted satisfactorily, re-tighten the screws on the side and replace the plug.

Getting Inside

Removing the Nose

The Pioneer 3-DX and –AT onboard computer sits just behind the robot's nose. And you may have to remove the nose to access the front SONAR array's gain adjustment pot. Two screws hold the nose to the front SONAR (or blank) array. The AT also has a screw at the bottom of the nose that attaches to the body; the DX's nose is hinged at the bottom.

Remove all nose retaining screws with the 3mm hex wrench supplied with your robot. Unlike earlier Pioneer 2 models, you do not have to remove the Gripper or the front bumper accessories.

Once loosened, the DX nose pivots down on a hinge. For the AT model, four pins along the nose's back edges guide it onto the front of the robot. Simply pry the nose out and away from the body.

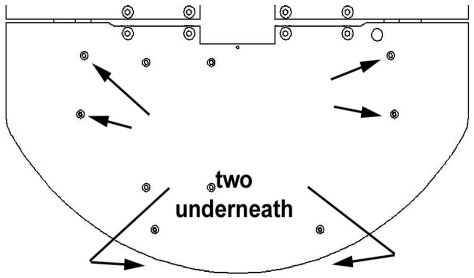


Figure 15. Remove indicated screws to access the front plate of Pioneer 3-DX and -AT robots.

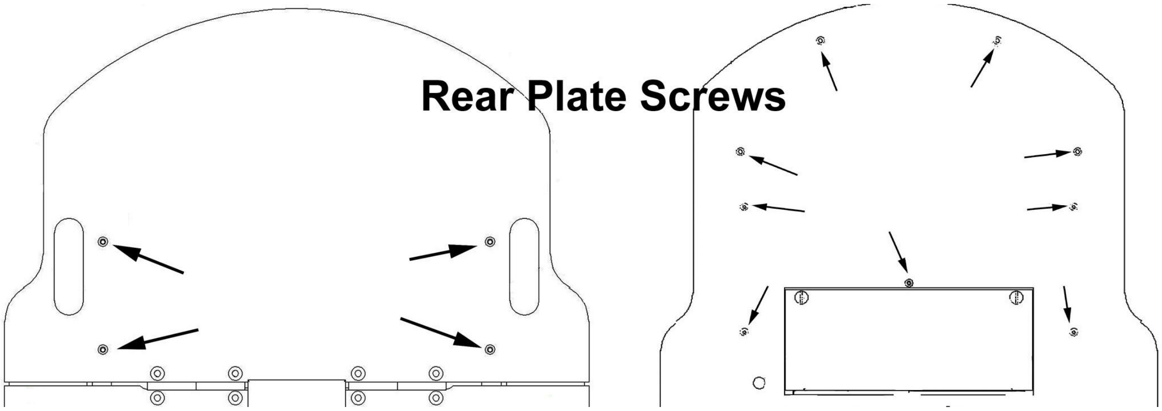


Figure 16. Remove indicated screws from Pioneer 3-DX or -AT rear deck to open plate.

Careful: The computer's hard-drive, fan, and speaker have attached wire harnesses that you need to relieve before completely detaching the nose from the body. We recommend unplugging the speaker wire and simply rotating the nose out of the way to access the onboard computer.

Opening the Deck

All Pioneer 3 robots have a center hinge in the deck which let you easily open and access internal components without completely removing the top plate. Simply remove the indicated 3mm screws shown in the Figures nearby from the section of the deck that you want to access. You may need to first remove any accessories that are bolted to the top plate through the indicated holes.

Remove the batteries BEFORE opening the robot.

Factory Repairs

If, after reading this manual, you're having *hardware* problems with your Pioneer 3 robot and you're satisfied that it needs repair, contact us:

```
http://robots.MobileRobots.com/techsupport
```

Tell us your robot's SERIAL NUMBER

Describe the problem in as much detail as possible. Also include your robot's serial number (IMPORTANT!) as well as name, email and mail addresses, along with phone and fax numbers. Tell us when and how we can best contact you (we will assume email is the best manner, unless otherwise notified).

Use authorized parts *ONLY*;
warranty void otherwise.

We will try to resolve the problem through communication. If the robot must be returned to the factory for repair, obtain a shipping and repair authorization code and shipping details from us first.

We are not responsible for shipping damage or loss.

Chapter 8 Appendix A

ROBOT MICROCONTROLLER PORTS & CONNECTORS

This Appendix contains pin-out and electrical specifications for the external and internal ports and connectors on the SH2-based microcontroller for the Pioneer 3, PeopleBot and PowerBot, including motor-power interface and User Control boards.

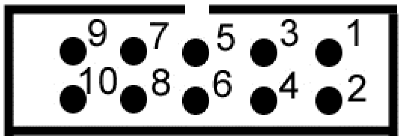


Figure 17. IDC connector

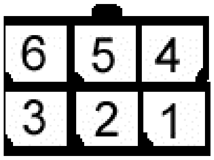


Figure 18. Mini-fit and micro-fit connector

Function	Uses	Connector Label on Board	Available for user custom use?
Digital inputs ID0-7	Gripper, dock, or user custom I/O	"USER IO/GRIPPER" Connector	Yes, if no gripper
Digital outputs OD0-7	Gripper, dock or user custom I/O	"USER IO/GRIPPER" Connector	Yes, if no gripper
Analog input AN0	Joystick speed control	"USER IO/GRIPPER" Connector and "JOYSTICK" Connector	
Analog input AN1	Joystick X axis	"JOYSTICK" Connector	
Analog input AN2	Joystick Y axis	"JOYSTICK" Connector	
Analog input AN3	Internal	n/a	
Analog input AN4	Available for user custom I/O	"TILT/ROLL" connector	Yes
Analog input AN5	Available for user custom I/O	"TILT/ROLL" connector	Yes
Analog input AN6	Gyro	"GYRO" connector	
Analog input AN7	Gyro	"GYRO" connector.	
Joystick button FB0	Joystick button	"JOYSTICK" Connector	
Joystick button FB1	Joystick button	"JOYSTICK" Connector	
Bumper switch inputs DB0-DB7	Bumpers	"BUMPERS" Connector	
IR inputs IR0-IR7	Infrared sensors on Peoplebot, or user custom inputs	"MOTORS" motor/power/encoder/IR connector.	Yes, if no IR sensors
HOST serial port	Software control of robot	User control panel, and "HOST" connector.	
AUX1-3 serial ports	Accessory or user connections	"AUX1" , "AUX2", "AUX3" serial port connectors	Yes, if no accessory

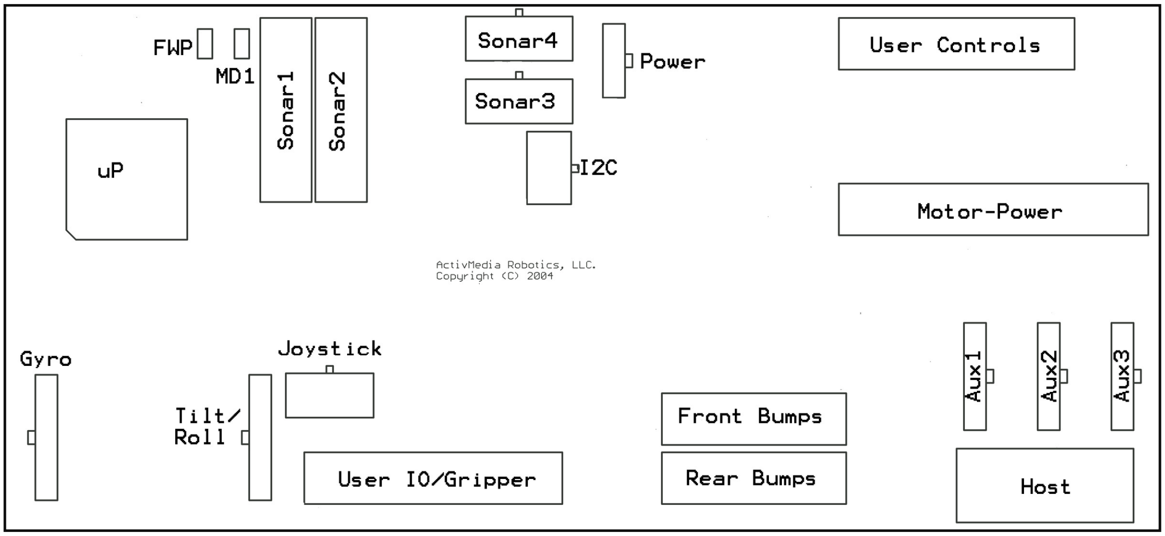


Figure 19. SH-2 Microcontroller Board Connector Locations

Main Power Input

The power connector is a 3-pin microfit socket that delivers 12 and 5 VDC to the microcontroller, including power ground, supplied by the Motor-Power board (see below).

Table 22. Microcontroller Power Connector

PIN	DESCRIPTION
1	12 VDC (battery)
2	GND
3	5VDC

Auxilliary Serial Ports

One DSUB-9 and three 5-position microfit sockets provide the HOST and AUX1, AUX2, AUX3 serial ports for the SH2 microcontroller. All are RS-232 compatible. The HOST port is shared with the User Control Panel's SERIAL port and is for ARCOS client-server and maintenance connections. The HOST serial connector also has signal lines for detecting an attached device (DTR pin 4) and for notifying the attached PC of low-power condition (DSR pin 6 and HRNG pin 9). The HOST serial connectors are wired DCE for direct connection (straight-through cable, not NULL-modem) to a standard PC serial port. See the nearby Tables for details.

The AUX1, AUX2 and AUX3 ports are extra RS-232 serial interfaces on the robot microcontroller. Data can be sent via the robot microcontroller using "TTY" and "GETAUX" commands (see Chapter 6).

The serial ports operate at any of the common data rates: 9.6, 19.2, 38.4, 57.8, or 115.2 kilobits per second, and with eight data bits, one stop bit, no parity or hardware handshaking.

AUX SERIAL Sockets:	5-position microfit (3mm, single row) header (male pins) Molex part #43650-0515
Mating connector:	5-pos microfit 3mm single row connector receptacle (female socket) with latch lock Molex part #43645-0500
	www.digikey.com part #WM1848-ND
	www.mouser.com part #538-43645-0500

Table 23. HOST serial ports on microcontroller and on User Control panel

PIN	SIGNAL	DESCRIPTION	PIN	SIGNAL	DESCRIPTION
1	nc		2	TXD	Signal out
3	RCV	Signal in	4	DTR	Input detects attached device and switches TxD and RxD into the uC
5	GND	Common	6	DSR	Output when microcontroller powered
7	nc		8	nc	
9	HRING	Set low to signal low battery voltage			

Table 24. Aux1, Aux2, and Aux3 serial ports (5-pos microfit sockets)

PIN	SIGNAL	DESCRIPTION	PIN	SIGNAL	DESCRIPTION
1	DTR	Aux1 only	2	TXD	Signal out
3	RCV	Signal in	4	DSR	Output active
5	GND	Common			

User I/O, Gripper and Automated Recharger Connector

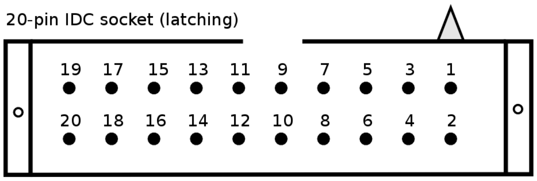
A 20-pin latching IDC socket on the microcontroller provides the digital IO, analog IO and power ports for user I/O or accessories. If the Pioneer Gripper or docking hardware is installed in a robot, the digital I/O ports used for those devices will not be available for user custom devices. Indicated ports (*) are shared on other connectors. Digital inputs are buffered and pulled high (digital 1); outputs are buffered and normally low (digital 0).

The states of the buffered digital input and output ports are provided to client software in the DIGIN and DIGOUT field of the standard information packet (SIP), and are included in the IO Packet as well. The state of digital outputs port may be set by sending the DIGOUT command (#30). The voltage on the AN0 analog input port is provided in the ANALOG field of the SIP if AN0 was selected with the ADSEL command, and is also included in the IO Packet as well. See Chapter 6, ARCOS, for details.

NOTE: AN0 is used by the Pioneer joystick if your robot was equipped with this option, and is also present on the Joystick connector. If used for a joystick it cannot be used for user custom I/O; use AN4 or AN5 instead (see below).

Socket:	20-position IDC socket (latching). 3M part #3428-6302
Mating connector:	20-position IDC connector. Crimp with 26-28AWG ribbon cable.
	MobileRobots part #CAB1003 (with cable: part #CAB1075)
	3M part #89120-0101
	www.digikey.com part #MKC20A-ND
	www.mouser.com part #517-8920

20-pin IDC socket (latching)



20-pin IDC connector & cable

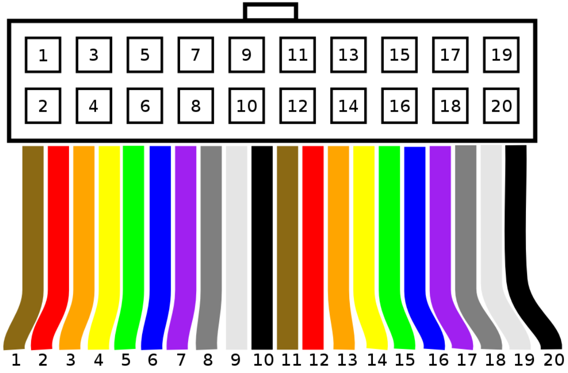


Table 25. User I/O – Gripper (20-pos latching IDC)

Pin	Signal	Description
1	OD0	DIGOUT bit 0; Gripper enable
3	OD1	DIGOUT bit 1; Gripper direction
5	OD2	DIGOUT bit 2; Lift enable
7	OD3	DIGOUT bit 3; Lift direction
9	ID4	DIGIN bit 4; Left paddle contact
11	ID5	DIGIN bit 5; Right paddle contact
13	ID6	DIGIN bit 6; "power good"
15	ID7	DIGIN bit 7; "overcharge"
17	AN0	Analog port 0
19	Vpp	Battery 12VDC < 1A

Pin	Signal	Description
2	ID0	DIGIN bit 0; Paddles open limit
4	ID1	DIGIN bit 1; Lift limit
6	ID2	DIGIN bit 2; Outer breakbeam IR
8	ID3	DIGIN bit 3; Inner breakbeam IR
10	OD4	DIGOUT bit 4; "inhibit"
12	OD5	DIGOUT bit 5; "deploy"
14	OD6	DIGOUT bit 6; User only
16	OD7	DIGOUT bit 7; User only
18	Vcc	5VDC < 1A
20	Gnd	Signal/power common

AN4, AN5 Analog Inputs ("TILT/ROLL")

The "TILT/ROLL" connector provides connections for two additional analog I/O inputs, AN4 and AN5, as well as an extra 5V power connection. These values are available in the IO Packet (see Chapter 6, ARCOS above for details). They are not used for any purpose by ARCOS itself.

Motors, Encoders Table 26. "tilt/roll" connector

Pin	Signal	Description
1	nc	
3	AN4	AN4

Pin	Signal	Description
2	VCC	5 VDC power
4	AN5	AN5

5	AGND	AN signal ground	6	GND	Power ground

Motors, Encoders and IRs

A 26-position latching IDC connector on the microcontroller connects the Microcontroller to the Motor-Power Board (see Appendix B). Line descriptions also can be found in the following Motor-Power Interface section.

Table 27. Motors, encoders, and IRs interface (26-pos latching IDC)

Pin	Signal	Description	Pin	Signal	Description
1	LPWM	Left motors PWM	2	LDIR	Left motors direction
3	RPWM	Right motors PWM	4	RDIR	Right motors direction
5	MEN	Motors enable	6	LEA	Left encoder channel A
7	E-STOP	E-Stop detect input	8	REA	Right encoder channel A
9	RPWR	Aux1 power enable	10	REB	Right encoder channel B
11	APWR	Aux2 power enable	12	LEB	Left encoder channel B
13	CHRG	Charge port detect	14	IR6	IR input bit 6
15	IR7	IR input bit 7	16	IR4	IR input bit 4
17	IR5	IR input bit 5	18	IR2	IR input bit 2
19	IR3	IR input bit 3	20	IR0	IR input bit 0
21	IR1	IR input bit 1	22	VBAT	Battery voltage
23	Gnd	Signal common	24	AN1*	Analog input
25	Gnd	Signal common	26	AN2*	Analog input

* Board versions C and earlier pin 24 HOST RI and pin 26 ground.

Joystick

An 8-position microfit socket provides signal lines for connection to the inductive joystick accessory. Indicated lines (*) are shared on other connectors.

NOTE: AN0 input is also present on the user IO connector.

Table 28. Joystick connector (8-pos microfit)

pin	signal	description	pin	signal	description
1	Vcc	5 VDC	2	FB0	Fire button 0
3	AN2	Turn Y-axis	4	AGND	Analog common
5	AN1	Translate X-axis	6	FB1	Fire button 1
7	*AN0	Max speed control	8		nc

Bumpers

Two 10-position latching IDC connectors provide general-purpose digital inputs, typically used for the robot's bumpers. All inputs are buffered and pulled high (digital 1).

Table 29. Bumper ports (10-pos latching IDC)

PIN	SIGNAL	DESCRIPTION	PIN	SIGNAL	DESCRIPTION
1	BP0	Bumper bit 0	2	BP1	Bumper bit 1
3	BP2	Bumper bit 2	4	BP3	Bumper bit 3
5	BP4	Bumper bit 4	6	BP5	Bumper bit 5
7	BP6	Bumper bit 6	8	BP7	Bumper bit 7
9	Gnd	Common	10	Gnd	Common

SONAR

Four connectors—two latching 10-pos IDC and two 10-pos microfits—provide signal and power for the four SONAR arrays SONAR1 through SONAR4, respectively.

Table 30. SONAR

PIN	SIGNAL	DESCRIPTION	PIN	SIGNAL	DESCRIPTION
1	A0	disc address	2	A1	disc address
3	A2	disc address	4	BINH	inhibits return signal
5	INIT	starts SONAR ping	6	VCC	5 VDC
7	VCC	5 VDC	8	SGND	Common
9	SGND	Common	10	ECHO	Goes high if echo threshold reached

User Control Board

A 16-position latching IDC connector provides interface with the User Control Panel board and functions. See description in a following section.

Table 31. User Control Panel interface

PIN	SIGNAL	DESCRIPTION	PIN	SIGNAL	DESCRIPTION
1	Vcc	5 VDC power	2	Vcc	5 VDC power
3	RST	RESET button	4	MOT	MOTORS button
5	RPWR	Radio power switch	6	APWR	Aux power switch
7			8	BZR	Buzzer PWM
9	PLED	Main power	10	SLED	Status
11	Vpp	Battery 12 VDC	12	Gnd	Signal/power common
13	Gnd	Signal/power common	14	HTXD	HOST serial transmit
15	HDSR	HOST serial enabled	16	HRCV	HOST serial receive

Gyro

The heading-correction gyro accessory attaches directly with the microcontroller through its respective 6-position microfit connector. Indicated lines (*) are shared on other connectors.

Table 32. Heading correction gyro connector

Pin	Signal	Description	Pin	Signal	Description
1	nc		**2**	VCC	5 VDC power
3	RATE	AN6	4	TEMP	AN7
5	AGND	Analog gnd	6	GND	Power ground

Chapter 9 Appendix B

MOTOR-POWER DISTRIBUTION BOARD

The Motor-Power Board provides power conversion and distribution to other robot components.

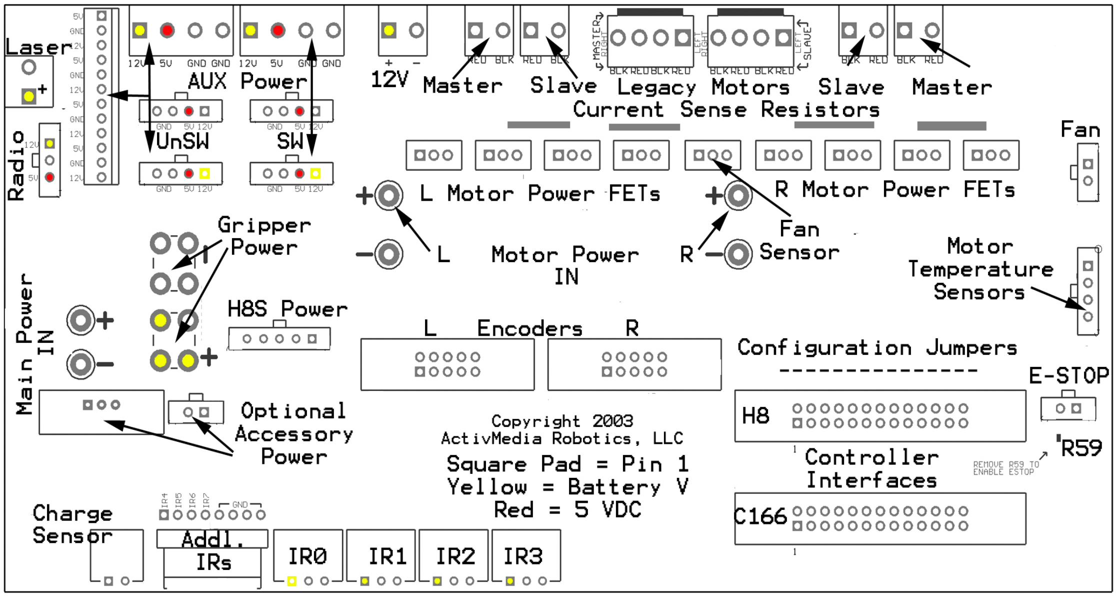


Figure 20. Pioneer 3 Motor-Power Board

Microcontroller Connection

Individual 26-pos IDC connectors and cables provide signal for the H8S- and SH2-based microcontrollers or for the legacy C166-based microcontrollers. A separate cable and connector provides for the SH2 microcontroller and SONAR power. Power and signal are shared on the C166 microcontroller connector.

Table 33. "H8S Power" connector, to microcontroller board (5-pos microfit)

PIN	FUNCTION	DESCRIPTION
1	Vbat	Battery power
2	Gnd	Power common
3	Vcc	5 VDC for SONAR
4	Vcc	5 VDC for SONAR
5	nc	No connection

Auxiliary and User Power Connectors

Various connectors provide conditioned 5 VDC @ 1.5A total and unconditioned battery power for the variety of accessories and custom user attachments. Some are AUX1 and AUX2 power switched from the User Control Panel.

The row of screw-down auxiliary user-power connectors along the top edge of the motor-powerboard make custom attachments easy.

(The 12-position latchlock connector to the left of the screw-down terminals was used in the past for some accessories.) Four-position microfit connectors also provide AUX power for standard accessories. AUX1 power was formerly called “Radio” power. AUX2 power is sometimes referred to or labeled as “Video” power as well.

Table 34. User Control Panel-switched AUX1 power connector (3-pos microfit)

Pin	Function	Description
1	Vpp	AUX1 (formerly radio) switched battery 12 VDC
2	Gnd	Power common
3	Vcc	AUX1 (formerly radio) switched 5 VDC
Socket on Board:		3-position microfit header (male pins, 3mm) Molex part #43650-0315
Mating connector:		3-position connector receptacle (female) with latch lock. Add 20-30 AWG terminal contact pins to crimp with wire.
		Molex part #436450300
		www.digikey.com part #WM1846-ND
		www.mouser.com part #538-43645-0300

Table 35. User Control Panel-switched and unswitched Aux power connectors (4-pos microfits and screw-down terminal blocks)

Pin	Function	Description
1	Vpp	Aux switched battery 12 VDC
2	Vcc	Aux switched 5 VDC
3	Gnd	Power common
4	Gnd	Power common
Socket on Board:		4-position microfit header (male pins, 3mm) Molex part #43650-0415
Mating connector:		4-position connector receptacle (female) with latch lock. Add 20-30 AWG terminal contact pins to crimp with wire.
		Molex part #436450-400
		www.digikey.com part #WM1847-ND
		www.mouser.com part #538-43645-0400

Table 36. User Power connector (12-pos latchlock; unswitched)

Pin	Function	Pin	Function
1	Vcc	7	Vcc
2	Gnd	8	Gnd
3	Vpp	9	Vpp

4	Vcc
5	Gnd
6	Vpp

10	Vcc
11	Gnd
12	Vpp

IR Signal and Power

Four connectors provide power and signal for fixed-range IR sensors. A separate connector provides signal path for an additional four IR sensors.

Table 37. IR power and signal connectors (3-pos microfits)

Pin	Function	Description
1	Vpp	Battery 12 VDC
2	IRn	Switching signal
3	Gnd	Power/signal ground

Table 38. Additional IR connector (8-pos latchlock 0.1 header)

Pin	Signal	Description
1-4	IR4-7	IR signals
5-8	GND	Signal common

Chapter 10 Appendix C

SPECIFICATIONS

	Pioneer 3-DX	Pioneer 3-AT
Physical Characteristics		
Length (cm)	44.5	50.1
Width (cm)	39.3	49.3
Height (cm)	23.7	27.7
Clearance (cm)	6.0	8.4
Clearance bumpers (cm)	3.5	5.4
Weight (kg)	9	14
Payload (kg)	25	15
Power		
Batteries 12VDC lead-acid (7 or 9 Ah)	3	3
Charge (watt-hours)	252	252
Run time (approx. hours)	8–10	4-6
with PC (approx. hours)	3-4	2-3
Recharge time low speed charger (1 battery)	6	6
High-Speed (3 batteries)	2.4	2.4
Mobility		
Wheels	2 foam-filled	4 pneumatic
tread	knobby	Wave (optional narrow indoor wheels also available)
diameter (mm)	195.3	220
width (mm)	47.4	75 (optional narrow indoor wheels also available)
Tire pressure	na	23 psi (1.6 bar)
Caster (mm)	75	na
Steering	Differential	Skid
Gear ratio	38.3:1	49.8:1
Swing (cm)	26.7	34
Turn (cm)	0	0
Translate speed max (mm/sec)	1,400	700
Rotate speed max (deg/sec)	300	140
Traversable step max (mm)	20	89
Traversable gap max (mm)	89	127
Traversable slope max (grade)	25%	40%
Traversable terrains	Wheel-chair accessible	Unconsolidated/loose/high friction (e.g. sand, gravel, dirt) *No carpets*!

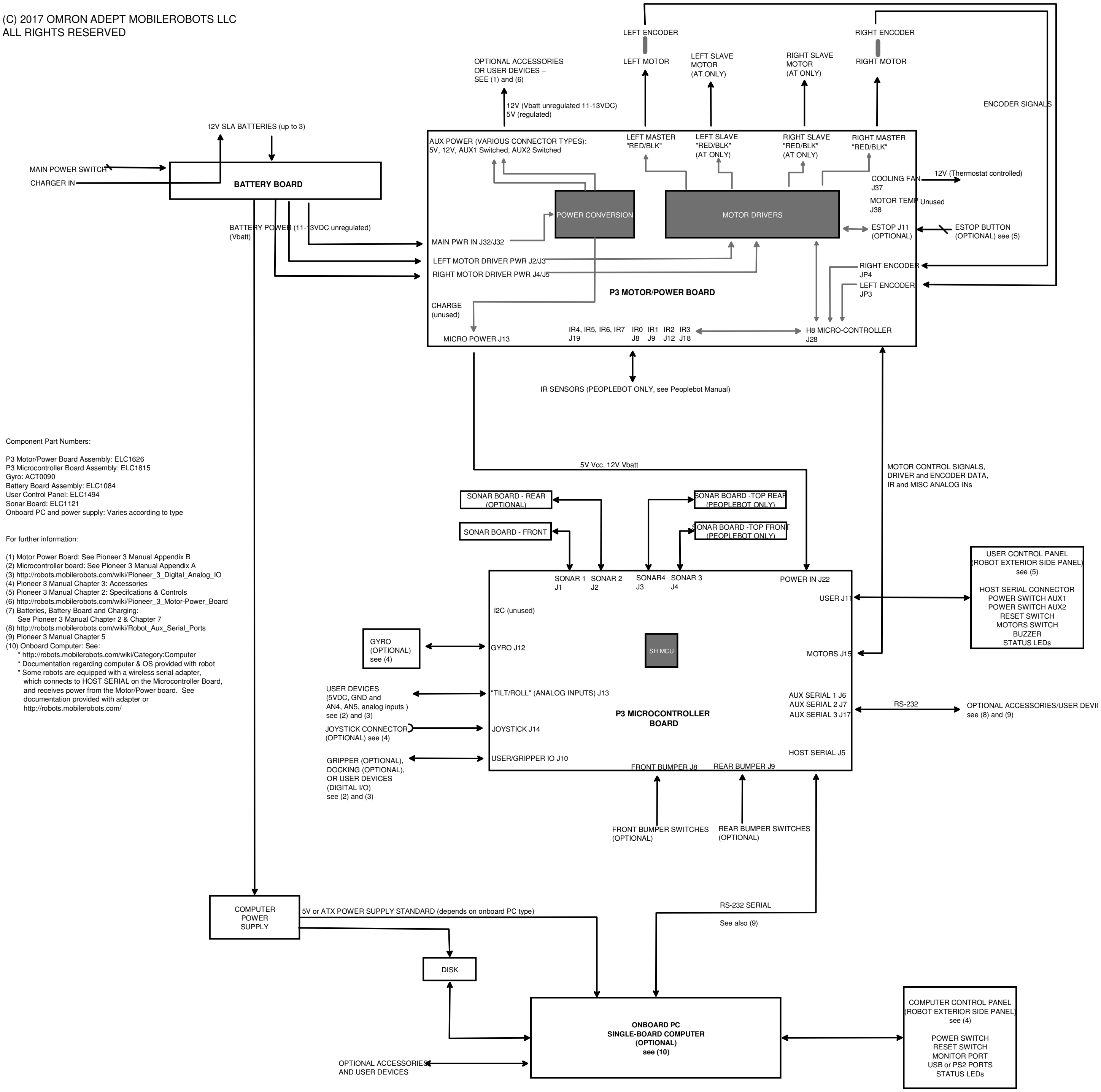
Sensors	Pioneer 3-DX	Pioneer 3-AT
SONAR Front Array (one each side, six forward @ 20° intervals)	8 standard	8 optional
Rear SONAR Array (one each side, six rear @ 20° intervals)	8 optional	8 optional
Bumpers	Optional	Optional
Motors		
Motors	Two 12V DC gearmotors	Four 12V DC gearmotors. Two motors on each side, linked by belt.
Gear ratio	38.3:1	49.8:1 (motor gearbox 38.3:1 plus belt pulleys)
Encoders	500-count quadrature optical encoder on each motor	500-count quadrature optical encoder on one motor on each side
Quadrature encoder count/wheel revolution	19,150 (200mm wheel)	24,900 (220mm wheel)
Quadrature encoder counts/mm robot linear distance travelled	128	138
Quadrature encoder counts/360° robot rotation	33,140	65,100
Ports and Controls		
Main Power Switch	Standard	Standard
Motor E-Stop Button	Optional	Standard
Aux1 Power Switch	5 & 12 VDC	5 & 12 VDC
Aux2 Power Switch	5 & 12 VDC	5 & 12 VDC
General purpose power (unswitched)	5 & 12 VDC	5 & 12 VDC
System Serial Control Connection	Standard DSUB-9 Port; USB-serial adapter optional	Standard DSUB-9 Port; USB-serial adapter optional
Motors Button	Standard	Standard
Microcontroller Reset Button	Standard	Standard
Digital and Analog IO	Microfit header in robot	Microfit header in robot

Chapter 11 Appendix D

SYSTEM INTERCONNECTION/WIRING DIAGRAM

PIONEER 3 INTERCONNECTION/WIRING DIAGRAM

REV A 4/7/2017 RH initial drawing


LEFT ENCODER
RIGHT ENCODER
LEFT MOTOR
LEFT SLAVE MOTOR (AT ONLY)
RIGHT SLAVE MOTOR (AT ONLY)
RIGHT MOTOR
OPTIONAL ACCESSORIES OR USER DEVICES -- SEE (1) and (6)
12V (Vbatt unregulated 11-13VDC)
5V (regulated)
ENCODER SIGNALS
12V SLA BATTERIES (up to 3)
MAIN POWER SWITCH
CHARGER IN
BATTERY BOARD
BATTERY POWER (11-13VDC unregulated)
(Vbatt)
AUX POWER (VARIOUS CONNECTOR TYPES):
5V, 12V, AUX1 Switched, AUX2 Switched
LEFT MASTER "RED/BLK"
LEFT SLAVE "RED/BLK" (AT ONLY)
RIGHT SLAVE "RED/BLK" (AT ONLY)
RIGHT MASTER "RED/BLK"
COOLING FAN J37
12V (Thermostat controlled)
MOTOR TEMP J38
Unused
POWER CONVERSION
MOTOR DRIVERS
ESTOP J11 (OPTIONAL)
ESTOP BUTTON (OPTIONAL) see (5)
MAIN PWR IN J32/J32
LEFT MOTOR DRIVER PWR J2/J3
RIGHT MOTOR DRIVER PWR J4/J5
RIGHT ENCODER JP4
LEFT ENCODER JP3
P3 MOTOR/POWER BOARD
CHARGE (unused)
MICRO POWER J13
IR4, IR5, IR6, IR7 J19
IR0 J8
IR1 J9
IR2 J12
IR3 J18
H8 MICRO-CONTROLLER J28
IR SENSORS (PEOPLEBOT ONLY, see Peoplebot Manual)
5V Vcc, 12V Vbatt
MOTOR CONTROL SIGNALS, DRIVER and ENCODER DATA, IR and MISC ANALOG INs
SONAR BOARD - REAR (OPTIONAL)
SONAR BOARD - FRONT
SONAR BOARD -TOP REAR (PEOPLEBOT ONLY)
SONAR BOARD -TOP FRONT (PEOPLEBOT ONLY)
SONAR 1 J1
SONAR 2 J2
SONAR4 J3
SONAR 3 J4
POWER IN J22
USER CONTROL PANEL (ROBOT EXTERIOR SIDE PANEL) see (5)
HOST SERIAL CONNECTOR
POWER SWITCH AUX1
POWER SWITCH AUX2
RESET SWITCH
MOTORS SWITCH
BUZZER
STATUS LEDs
USER J11
I2C (unused)
GYRO (OPTIONAL) see (4)
GYRO J12
SH MCU
MOTORS J15
USER DEVICES (5VDC, GND and AN4, AN5, analog inputs) see (2) and (3)
"TILT/ROLL" (ANALOG INPUTS) J13
AUX SERIAL 1 J6
AUX SERIAL 2 J7
AUX SERIAL 3 J17
RS-232
OPTIONAL ACCESSORIES/USER DEVI
see (8) and (9)
P3 MICROCONTROLLER BOARD
JOYSTICK CONNECTOR (OPTIONAL) see (4)
JOYSTICK J14
GRIPPER (OPTIONAL), DOCKING (OPTIONAL), OR USER DEVICES (DIGITAL I/O) see (2) and (3)
USER/GRIPPER IO J10
HOST SERIAL J5
FRONT BUMPER J8
REAR BUMPER J9
FRONT BUMPER SWITCHES (OPTIONAL)
REAR BUMPER SWITCHES (OPTIONAL)
COMPUTER POWER SUPPLY
5V or ATX POWER SUPPLY STANDARD (depends on onboard PC type)
RS-232 SERIAL
See also (9)
DISK
ONBOARD PC SINGLE-BOARD COMPUTER (OPTIONAL) see (10)
COMPUTER CONTROL PANEL (ROBOT EXTERIOR SIDE PANEL) see (4)
POWER SWITCH
RESET SWITCH
MONITOR PORT
USB or PS2 PORTS
STATUS LEDs
OPTIONAL ACCESSORIES AND USER DEVICES

Component Part Numbers:

P3 Motor/Power Board Assembly: ELC1626
P3 Microcontroller Board Assembly: ELC1815
Gyro: ACT0090
Battery Board Assembly: ELC1084
User Control Panel: ELC1494
Sonar Board: ELC1121
Onboard PC and power supply: Varies according to type

For further information:

(1) Motor Power Board: See Pioneer 3 Manual Appendix B
(2) Microcontroller board: See Pioneer 3 Manual Appendix A
(3) http://robots.mobilerobots.com/wiki/Pioneer_3_Digital_Analog_IO
(4) Pioneer 3 Manual Chapter 3: Accessories
(5) Pioneer 3 Manual Chapter 2: Specifcations & Controls
(6) http://robots.mobilerobots.com/wiki/Pioneer_3_Motor-Power_Board
(7) Batteries, Battery Board and Charging:
See Pioneer 3 Manual Chapter 2 & Chapter 7
(8) http://robots.mobilerobots.com/wiki/Robot_Aux_Serial_Ports
(9) Pioneer 3 Manual Chapter 5
(10) Onboard Computer: See:
- http://robots.mobilerobots.com/wiki/Category:Computer
- Documentation regarding computer & OS provided with robot
- Some robots are equipped with a wireless serial adapter, which connects to HOST SERIAL on the Microcontroller Board, and receives power from the Motor/Power board. See documentation provided with adapter or http://robots.mobilerobots.com/

Chapter 12 Warranty & Liabilities

Your Pioneer 3 robot platform is fully warranted against defective parts or assembly for one year after it is shipped to you from the factory. Accessories are warranted for 90 days. Use only Omron Adept MobileRobots authorized parts or warranty void. This warranty also explicitly *does not include* damage from shipping or from abuse or inappropriate operation, such as if the robot is allowed to tumble or fall off a ledge, or if it is overloaded with heavy objects.

The developers, marketers and manufacturers of Omron Adept MobileRobots Pioneer products shall bear no liabilities for operation and use of the robot or any accompanying software except that covered by the warranty and period. The developers, marketers or manufacturers shall not be held responsible for any injury to persons or property involving Omron Adept MobileRobots products in any way. They shall bear no responsibilities or liabilities for any operation or application of the robot, or for support of any of those activities. And under no circumstances will the developers, marketers or manufacturers of Omron Adept MobileRobots products take responsibility for support of any special or custom modification to Omron Adept MobileRobots platforms or their software.

Index