

# **PARGPS User's Guide**

Manual Release 01/01/04  
Copyright (c), Symmetric Research, 2004

Web: [www.symres.com](http://www.symres.com)

## LIMITED WARRANTY

### *WHAT IS COVERED*

Symmetric Research warrants its PARGPS product will be free from defects in workmanship and materials for one year from the date of original purchase.

### *WHAT SR WILL DO*

Symmetric Research will repair or replace defective PARGPS systems covered under this warranty at no cost to the customer other than shipping. The customer is responsible for shipping to SR manufacturing facilities.

### *WHAT IS NOT COVERED*

Symmetric Research does not warrant the PARGPS product for use with customer provided power supplies. Incorrectly connecting power may permanently damage the system.

Furthermore, PARGPS systems that have been customer modified are also not covered under this warranty.

**Symmetric Research will at its discretion determine when any returned equipment has been run from incorrect power supplies and is not covered by the terms of this warranty.**

Symmetric Research is not liable for any loss, damage, or inconvenience, including direct, special, incidental, or consequential damages, resulting from the use or inability to use the PARGPS product.

## TABLE OF CONTENTS

Chapter 1: Introduction.....	1
Chapter 2: Installation.....	2
Chapter 3: Diagnostic and Utility Programs .....	7
Chapter 4: Library Functions .....	9
Appendix A: Electrical Specs .....	16
Index .....	18

## INTRODUCTION

The Symmetric Research PARGPS timing module provides accurate time stamping for data acquired with the SR PARxCH A/D 24 bit A/D boards. Typical timing accuracy is 10 microseconds.

Based on the Trimble ACE-III GPS receiver, the SR PARGPS board provides both time and location information on an ongoing real time basis as data is acquired. It uses industry standard RS232 National Marine Electronics Association (NMEA) messages for location and coarse timing. Precision timing is provided by the highly accurate pulse-per-second (PPS) signal. With its direct connection to the PARxCH digital inputs, the PARGPS board provides excellent time stamping with the least dependence on PC interrupt latency.

The PARGPS board sits outside the PC in a vinyl covered steel enclosure that stacks nicely with the PAR4CH. Alternatively, for PC104 applications, an adapter plate is available to connect the PARGPS and PAR4CH to a PC104 stack. A standard DB9 serial port cable to COM1 or COM2 carries the NMEA messages from the PARGPS to the PC, while the PPS signal is fed directly to the digital I/O connector of the PARxCH for processing by the A/D system.

Included with the system is full software support for WinNT/2000/XP and Linux. The high level PARxCH acquisition applications such as scope and simp have been upgraded for GPS support, along with a new DAT file format for storing ongoing time and location information along with the acquired data. Additional utilities are included for various operations with the time information. As with all SR products, the included circuit diagrams, low level function libraries and source code allow users to customize the system to their specifications. The WinNT/2000 and Linux software features true kernel mode drivers for good performance and reliable operation.

Other items included with the system are a cable for sharing power with the PARxCH, a cable for connecting the PPS signal to the PARxCH, a cable for connecting to the PC serial port, a 6 meter antenna, and circuit diagrams. This is everything needed to be up and running right away.

We hope you enjoy using the SR PARGPS

## **INSTALLATION**

You must have an SR PARxCH A/D acquisition board to use the PARGPS system. The PARxCH and PARGPS work together. Installation of the PARGPS is straightforward. First, install the PARxCH as described in the PARxCH User's Guide. Then you'll need to daisy chain power with the PARxCH, connect to the PARxCH digital I/O header, connect to the PC serial port, connect the antenna, and install the software. The individual steps in more detail are:

### **CONNECTING POWER**

The PARGPS requires power independently of the PC. Connect the wall transformer supplied with the PARxCH to one of the 2.1mm connectors on the back of the PARGPS unit. Then use the 12" black cable supplied with the PARGPS to daisy chain power from the extra 2.1mm connector on the PARGPS back to the PARxCH.

Once plugged in, you can verify the wall transformer is on by checking the green LED near the power connector on the back of the PARGPS. If it is on, your wall transformer is plugged in and powered on.

The PARGPS can be powered with a wide range of wall transformers. Any AC or DC wall transformer with a voltage between 9 and 24 volts is acceptable as long as it has a 2.1mm, center-plus plug. If you have a choice, chose a voltage nearer to 9 volts rather than 24. Running at lower power supply voltages will reduce the heat dissipation of the internal regulators. If you are running the PARGPS outside of its enclosure in a PC104 application, you can connect power via the 4 position Molex connectors J402 or J403 instead. Consult the circuit diagrams and board legend for proper connections.

The PARGPS is protected with diodes so it is unlikely incorrectly connected power supplies will damage it. However, the PARGPS is not protected from excessive overvoltages. Connecting either AC or DC power that is badly out of range may damage the board. Customers using their own power supplies should be aware that they are responsible for providing the correct voltages. Please read the limited warranty at the beginning of this manual.

### **CONNECTING THE PPS SIGNAL TO THE PARXCH**

Use the included short ribbon cable with 15 pin female D-shell connectors for connecting the PPS signal from the PARGPS board to the digital I/O connector on the right front of the

PARxCH A/D. This connection is easiest to make when the PARGPS unit is stacked above or below the PARxCH.

### **CONNECTING TO THE PC SERIAL PORT**

Use the included 6 ft molded 9 pin D-shell cable for connecting to the PC serial port. Connect the female end of this cable to the D-shell connector on the back of the PARGPS, and the male end to a serial port connector on your PC. It doesn't matter whether you connect to COM1 or COM2, but make a note of which one you select.

### **CONNECTING THE ANTENNA**

On the back left of the PARGPS unit is a small coax connector. Connect the supplied antenna to this connector. The other end of the coax cable contains the active part of the GPS antenna. This active end of the **antenna must have access to the open sky** around the equator in order for the system to work. The antenna will not work if buried under layers of concrete or in basements. More sophisticated antennas for difficult environments are available. Contact SR for information about companies that can supply such antennas.

After following the 4 steps above, the red LED on the front of the PARGPS unit should be toggling on and off at a 1 second rate. If not, check power and the green LED on the back of the PARGPS. Note that the red LED **does not** indicate GPS satellites have been acquired. The system emits a PPS signal even with no satellite lock. Only by running the diagnostic software can you determine the PPS quality and number of satellites the system is currently seeing.

### **INSTALLING THE SOFTWARE**

A CD-ROM is included with the PARGPS. Software for each supported operating system can be found in the respective directory. Each OS subdirectory includes a compressed PKZIP or Linux gzipped tar format file, and an installation batch or script file. To install the PARGPS software, change to the OS subdirectory appropriate for your computer and run the install batch script. This will automatically unzip and create the default \sr\parxch and \sr\pargps or Linux /usr/local/sr/parxch and /usr/local/sr/pargps directory structure on your hard disk.

We highly recommend that you keep only one copy of the software on your hard disk and that you use the default directory structure. This makes maintenance and installing upgrades easier. See the readme.txt files on the CD for more information.

For WinNT/2000/XP and Linux installations, besides copying the software to your hard disk, you must also install a kernel mode device driver for both the PARxCH and the PARGPS.

Administrator or root permissions are required for this step, but once the driver is installed, ordinary users can use it freely.

To install the PARxCH and PARGPS kernel mode device driver, change to the respective driver subdirectory on your hard disk and run the indriver program giving the name of the device driver as an argument. Run indriver with no arguments to see a list of valid driver names. You can remove the device driver using this name and the respective rmdriver program. When choosing the PARxCH driver name, you must pick the one which corresponds to the parallel port you are using for the A/D board.

For example, to install the PARxCH and PARGPS drivers typical usage would be:

```
> cd \sr\parxch\driver
> indriver SrParXch378 PAR4CH
> cd \sr\pargps\driver
> indriver SrParGps
```

To remove them use:

```
> cd \sr\parxch\driver
> rmdriver SrParXch378
> cd \sr\pargps\driver
> rmdriver SrParGps
```

If you forget the exact assigned device driver name, you can find it on a list of installed drivers provided by the OS. For WinNT, open Devices icon from the Control Panel and look for driver names starting with Sr. For Win2000, open the Device Manager and look for drivers of the SR Precision Instrumentation class. For Linux, run /sbin/lsmmod. For additional information on installing the device driver, please refer to the readme.txt files in the both driver subdirectories.

If you have the PARxCH and PARGPS powered on and correctly connected, you can verify correct operation by running the diag programs. First run the PARxCH diag program in the parxch diag directory to test the A/D board and its connections. Then run the PARGPS diag program located in the pargps diags directory to test the PARGPS and its connections. Both diag programs are designed to be run from a command prompt. Follow the on screen messages for more information.

For programmers, the core part of the software supplied with the system is a collection of low level functions that are required to communicate with the PARGPS. For Windows systems, these functions can be linked in statically or used dynamically from the Dynamic Link Library pargps.dll. Programs using pargps.dll must be able to find it at run time or Windows will give an error message. The best way to inform the system of the location of pargps.dll is to add its location to your execution path. The following command can be executed from the command line or added to your autoexec.bat file:

```
> set path=%path%;\sr\pargps\lib
```

For Linux systems, the core low level functions can also be linked in statically or used dynamically from the shared library `pargps.so`. Programs using the shared library should set the `LD_LIBRARYPATH` environment variable so the library can be found at runtime. One way to do this is by executing the following command line or adding it to your `.profile` file:

```
LD_LIBRARYPATH=$ LD_LIBRARYPATH:/usr/local/sr/pargps/lib ; export \  
$LD_LIBRARYPATH
```

Included with the software are many `readme.txt` files and source code files with comments. We encourage you to refer to these for more information about the software.



**INSTALLATION CHECKLIST:**

- \* Install the PARxCH as described in the PARxCH User's Manual and run the PARxCH diagnostic program to be sure it is working correctly.
- \* Plug the wall transformer from the PARxCH into the PARGPS and daisy chain power over to the PARxCH. Check that the green LED on the PARGPS back panel is on.
- \* Connect the supplied 9 pin D-shell cable from the PARGPS to your PC serial port.
- \* Connect the supplied 15 pin D-shell ribbon cable between the PARxCH digital I/O connector and the PARGPS.
- \* Connect the supplied antenna.
- \* Run the PARGPS diagnostic program in the pargps diags directory. This will show whether the PARxCH and PARGPS drivers are installed and talking with each other. It also verifies that the PPS signal is being received and passed to the PARxCH and that the NMEA messages are coming in over the serial port. Note that it may take up to 10 minutes after 3 satellites are found for valid location information to appear in the NMEA strings.
- \* Try out the simp program to acquire some data and see the system work

## DIAGNOSTIC AND UTILITY PROGRAMS

The SR PARGPS comes with diagnostic and utility programs you can run immediately after installing the system. These programs will help you become more familiar with the system. The source code is included for those wanting to modify these programs for custom applications.

For more details, refer to the readme.txt files in the diags, currtime and utils directories and comments in the source files.

### **diag**

This is a simple diagnostic program, located in the diags directory, that lets you verify the PARGPS and PARxCH are properly connected and functioning correctly. It tests that both the PARGPS and PARxCH drivers are properly installed and checks the PPS signal and NMEA serial messages.

Diag takes up to five command line arguments. These specify the names of the PARGPS and PARxCH drivers, which serial port to use for the NMEA messages, which parallel port mode to use, and which PARxCH device to use. The arguments can be given in any order and arguments matching the provided default can be omitted as long as at least one argument is given. Run diag with no arguments for a usage message and a list of valid argument values.

### **currtime**

This is a command line program that displays the current GPS time. An active GPS program such as PARGPS diag or one of the PARxCH acquisition programs with GPS enabled must be running for currtime to work correctly. These active GPS programs store GPS time information in the device driver for currtime to retrieve and display. A GUI version of this program, called GpsTime, is available in the vbasic directory.

### **sethdrtm**

This is a utility program to update the start time in the header portion of a .DAT file based on the GPS data included in the body of the .DAT file. The PARxCH scope and simple acquisition programs can be used to generate .DAT files. The sethdrtm program is located in the utils directory.

**scope,simp,dat2asc**

These programs come with the PARxCH software, but are mentioned here since you will mostly likely want to use them when working with the PARGPS. Scope and simp are acquisition programs. When running them with the PARGPS, be sure to enable GPS and select the .DAT output file format. This can be done from the scope menus or the simp .ini file. Dat2asc is a utility program that converts the binary .DAT file to a simple text format.

## LIBRARY FUNCTIONS

The library functions are at the core of the software supplied with the SR PARGPS. They allow users to control board operation from high level languages without having to know the low level details of how the system operates. These functions can be statically linked to C programs for any OS. In addition, they are available as a Dynamic Link Library (DLL) under Windows and as a shared library (.so) under Linux. When used as a DLL, these functions can be called from other high level programming languages such as Microsoft Visual Basic. This chapter covers usage from programming environments like Visual C. **Remember that the PARGPS must be used in conjunction with the PARxCH and GPS data can not be acquired without using the PARxCH functions.**

The outline of how to use the PARxCH and PARGPS library functions is fairly simple. First call both the PARxCH and PARGPS Open functions to open the drivers and perform various initialization steps. Then call the PARxCH AttachGps function so the two drivers can properly interact with each other. Once the boards have been initialized, call both Start functions so the PARGPS begins checking for PPS signals and NMEA serial data and the PARxCH begins acquiring data. Then use ReadDataWithGpsMark, ReadPpsData and ReadSerialData to move the analog, PPS and serial data into PC memory arrays which can be displayed or saved to the hard disk. The analog data comes from the PARxCH hardware FIFO and while the PPS and serial data come from the PARGPS software FIFO. When you are done, call both Stop and Close functions to stop acquiring data and close the drivers.

The PARGPS library also includes several helper functions for working with the NMEA message strings and various representations of time.

When using the PARGPS library functions, be sure to include the header file pargps.h in any C source code. The prototypes in this file make sure the correct parameters are passed to the functions. You will also find the defined constants in pargps.h useful for making your programs more readable. When using dynamic linking, make sure the pargps.dll or pargps.so library is on your Windows execution path or Linux LD\_LIBRARY path so it can be found at run time.

The following is a discussion of each PARGPS library function. Refer to the PAR4CH program simp4ch.c for an example of how to use the library functions.

**OPEN AND CLOSE DEVICE**

C usage:

```
#include "pargps.h"

DEVHANDLE ParGpsOpen(
    char* ParGpsName,
    int SerialPortNumber,
    int InterruptMode,
    int *Error
);

int ParGpsClose( DEVHANDLE handle );
```

The Open function opens the PARGPS device driver and provides a device handle that is passed to the remaining library functions. It also initializes the PARGPS board. Open should be the first library function called.

It returns a valid device handle on success and the value BAD\_DEVHANDLE otherwise. If the optional Error argument is provided, it is filled with an error code giving more detail about why the open failed. See pargps.h for a list of the possible error codes. Pass NULL to ignore this parameter.

The ParGpsName parameter identifies the device driver. The SerialPortNumber parameter controls which serial port the PARGPS library uses to receive NMEA messages. The InterruptMode should typically be set to INTERRUPT\_ALTERNATING. Use the defined constants in pargps.h to specify these parameters.

The Close function cleanly shutdowns the device driver. It should be the last library function called. In some cases, it may be called implicitly at program termination. However, you should not rely on this.

**PARxCH ATTACH AND RELEASE GPS**

C usage:

```
#include "pargps.h"
#include "parxch.h"

int ParXchAttachGps(
    DEVHANDLE ParXchHandle,
    DEVHANDLE ParGpsHandle,
    int *Error
);

int ParXchReleaseGps( DEVHANDLE ParXchHandle );
```

The AttachGps function in the PARxCH library takes a device handle to both the PARxCH and the PARGPS and establishes a connection between the two drivers so they can interact. This interaction is required since the PARGPS PPS signal is presented to the PC via the interrupt associated with the PARxCH parallel port and both drivers need to know when that occurs.

It returns 1 on success and 0 otherwise. If the optional Error argument is provided, it is filled with an error code giving more detail about why the attach failed. See parxch.h for a list of the possible error codes. Pass NULL to ignore this parameter.

The ReleaseGps function removes the connection between the two drivers and should be called before the drivers are closed.

## START AND STOP EXECUTION

C usage:

```
#include "pargps.h"

void ParGpsStart( DEVHANDLE handle );

void ParGpsStop( DEVHANDLE handle );
```

After successfully opening the device driver and initializing the PARGPS board, the next step is to start checking for the PPS signal and the serial NMEA messages by calling Start.

Once the PARxCH and PARGPS are started you can use the ReadPpsData and ReadSerialData functions to transfer the GPS data from the PARGPS software FIFO over to the PC. Of course, you will likely also want to use the ParXchReadDataWithGpsMark to transfer the PARxCH analog data from the PARxCH hardware FIFO over to the PC.

After you are done acquiring, call the Stop function and then the Close function to close the driver.

**READ DATA**

C usage:

```
#include "pargps.h"

unsigned int ParGpsReadPpsData(
    DEVHANDLE handle,
    void *PpsValues,
    unsigned int PpsNvalues
    int *Error,
    );

unsigned int ParGpsReadSerialData(
    DEVHANDLE handle,
    void *SerialValues,
    unsigned int SerialNvalues
    int *Error,
    );
```

ReadPpsData and ReadSerialData each copy their respective types of data from the PARGPS software FIFO to the PC where it is available for further processing. Although a void\* type is used in the function declaration, the parameter should be an array of PPSDATA or SERIALDATA structures, respectively. See pargps.h for a definition of these structures.

To use these functions, you need an array to hold the PPS or serial data values and the dimension of that array. Each of these functions reads as much data as possible into its array up to Nvalues and returns the number of points actually read. This may be zero if no values of that type are ready yet.

The optional Error parameter is filled with 0 if all is well and with an error code if something has gone wrong. See pargps.h for a list of the possible error codes. You can ignore this parameter by passing NULL.

A typical code fragment for using ReadPpsData and ReadSerialData would be:

```
/* Declare arrays to contain acquired values. */
PpsValues = (PPSDATA *)malloc( Npps * sizeof( PPSDATA ) );
SerValues = (SERIALDATA *)malloc( Nser * sizeof( SERIALDATA ) );

/* Open driver and start execution ... */
```



```

/* Acquire and process data. */
while ( 1 ) {
    Nremain = Nvalues;
    NextPt = 0;

    /* Get Nvalues worth of data. */
    while ( Nremain > 0 ) {
        Newpts = ParXchReadDataWithGpsMark(
                                ParXchHandle,
                                &Values[NextPt],
                                Nremain,
                                &ErrPts );
        NewPps = ParGpsReadPpsData(ParGpsHandle,
                                &PpsValues[NextPtPps],
                                NremainPps,
                                &ErrPps );
        NewSer = ParGpsReadSerialData(
                                ParGpsHandle,
                                &SerValues[NextPtSer],
                                NremainSer,
                                &ErrSerial );
        /* Update Nremains for New points read ... */

        /* Check for errors. */
        if ( ErrPps != ErrSerial != PARGPS_ERROR_NONE )
            Error("Error %s\n", PARGPS_ERROR_MSG[Err]);
    }

    /* Do something with data. */
    SaveValuesToDisk( Values, PpsValues, SerialValues );
}

```

See simp.c in the PARxCH simple directory for complete code listings showing how to use these functions.

**COUNTER FREQUENCY AND LIBRARY REV**

C usage:

```
#include "pargps.h"

int ParGpsGetCounterFrequency( DEVHANDLE Handle, long *Freq );

void ParGpsGetRev( int *Rev );
```

GetCounterFrequency fills the Freq argument with information about how often the 64 bit counter is incremented. This counter, maintained by the OS, is used to mark the PPS and data sample times. So its frequency must be known in order to properly convert counter values into times.

GetRev fills the Rev argument with an integer representing the current revision number of the PARGPS library. For example, a value of 201 indicates version 2.01.

## APPENDIX A: ELECTRICAL SPECS

### POWER SUPPLY REQUIREMENTS:

The PARGPS system requires a minimum off board power supply of at least 9vdc or vac at 130ma of current for reliable operation. The maximum off board power supply is approximately 24 volts.

A 9vdc 500ma unregulated wall transformer is typically used with the system. When loaded at 130ma, this will supply more than 12vdc of unregulated power. More than enough to adequately power the PARGPS.

Many users will be supplied with a 2.1mm daisy chain cable so a single wall transformer can be shared between the PARGPS and another SR product. The two 2.1mm connectors on the back panel of the PARGPS enclosure are in parallel. See the circuit diagrams for more details.

AC power is acceptable for off board power and will be rectified by the on board circuitry. 9vac wall or chassis mount transformers are perfectly fine with no degradation in PARGPS performance.

On the board itself, off board power is rectified and regulated to +5vdc as required by the on board circuitry. Running at higher off board voltages will increase the heat dissipation in the on board regulator. Use off board power supplies of 12 volts or less to minimize power dissipation.

### SERIAL CABLE REQUIREMENTS:

The NMEA messages from the GPS receiver are communicated to the PC over an RS232 serial port. The SR PARGPS system has been designed to be used with serial cables that have all 9 wires connected straight through. The serial cable supplied with the system meets this requirement.

Serial cables that do not include all 9 wires but only the RX/TX pair or are "null modem" will not work. Do not cross RX/TX and make sure RTS and ground are connected. See the circuit diagram for more details.

In addition to NMEA messages, the GPS receiver generates a PPS signal, which is available on a DB15 connector on the PARGPS front panel. This PPS signal is designed to be connected from the PARGPS front panel to the digital IO connector on one of the SR A/D

boards with the supplied DB15 ribbon cable. The length of the supplied ribbon cable works best if the PARGPS is stacked above or below the SR A/D board.

**ACCURACY OF PPS SIGNAL:**

The PPS signal ultimately generates an interrupt to the PC. We have found typical PC interrupt latency to be about 10 microseconds. However, this latency may be longer in some cases if the system is undergoing heavy interrupt activity. Heavy use of interrupts by other drivers and programs may degrade PPS performance.

Monitor the regularity of the 64 bit PPS count interval spacing as recorded in a DAT file to determine the impact of interrupts on your system. If the number of counts from one PPS mark to another deviates wildly, then your system is suffering exceptional interrupt activity and you should investigate.

## INDEX

### A

ASCII Conversion.....	8
Antenna	
Connecting.....	3

### C

Cable Requirements.....	16
Currtime.....	7

### D

DAT File.....	7, 8
Device driver	
Installation.....	3
Name.....	4, 10
Removal.....	4
Diag.....	3, 4, 7

### E

Electrical specs.....	16
-----------------------	----

### H

Hardware installation.....	2
----------------------------	---

### I

Installation.....	2
Checklist.....	6
Device Driver.....	3
Hardware.....	2
Software.....	3
InterruptMode.....	10
Introduction.....	1

### K

Kernel mode driver.....	See Device driver
-------------------------	-------------------

### L

LED	
Green.....	2, 6
Red.....	3

### Library functions

Overview.....	9
ParGpsClose.....	10
ParGpsGetCounterFrequency.....	15
ParGpsGetRev.....	15
ParGpsOpen.....	10
ParGpsReadPpsData.....	13
ParGpsReadSerialData.....	13
ParGpsStart.....	12
ParGpsStop.....	12
ParXchAttachGps.....	11
ParXchReadDataWithGpsMark.....	12, 13
ParXchReleaseGps.....	11
Library path, setting.....	4

### N

NMEA Messages.....	1, 6, 7, 9, 10, 16
--------------------	--------------------

### P

ParGpsName.....	See Device driver name
Power supply	
Connecting.....	2
Requirements.....	16
PPS Signal	
Accuracy.....	17
Checking.....	6, 7
Connecting.....	2
Data.....	9, 13
Meaning.....	1, 3

### S

Satellites.....	3
Scope.....	8
Serial Port	
Connecting.....	3, 16
SerialPortNumber.....	10
Sethdrtm.....	7
Software installation.....	3

### T

Trimble.....	1
--------------	---