**CHAPTER 1**

**1.GPS Based Vehicle Travel Monitor**

* 1. **Introduction**

A **GPS navigation device** is a device that receives Global Positioning System (GPS) signals to determine the device’s location on Earth. GPS devices provide latitude and longitude information, and some may also calculate [altitude](http://en.wikipedia.org/wiki/Altitude), although this is not considered sufficiently accurate or continuously available enough (due to the possibility of signal blockage and other factors) to rely on exclusively to pilot aircraft. GPS devices are used by the military, by aircraft pilots, by sailors, and for recreational purposes by the public.

GPS devices may have capabilities such as:

* maps, including streets maps, displayed in human readable format via text or in a graphical format,
* [turn-by-turn navigation](http://en.wikipedia.org/wiki/Turn-by-turn_navigation) directions to a human in charge of a vehicle or vessel via text or speech,
* directions fed directly to an [autonomous vehicle](http://en.wikipedia.org/wiki/Autonomous_car) such as a robotic probe,
* [traffic congestion maps](http://en.wikipedia.org/wiki/Traffic_congestion_map) (depicting either historical or real time data) and suggested alternative directions,
* Information on nearby amenities such as restaurants, fueling stations, and [tourist attractions](http://en.wikipedia.org/wiki/Tourist_attraction).

**GPS may be able to answer:**

* roads or paths available,
* roads or paths that might be taken to get to the destination,
* if some roads are busy (now or historically) the best route to take,
* The location of food, fuel or other needs,
* The shortest route between the two locations.

**1.2 Consumer applications**

Consumer GPS navigation devices include:

* Dedicated GPS navigation devices
* GPS modules that need to be connected to a computer to be used
* GPS loggers that record trip information for download. Such [GPS tracking](http://en.wikipedia.org/wiki/GPS_tracking) is useful for trailblazing, mapping by hikers and cyclists, and the production of [geo coded photographs](http://en.wikipedia.org/wiki/Geocoded_photograph).
* Converged devices, including GPS Phones and [GPS cameras](http://en.wikipedia.org/wiki/Auto-geotagging), in which GPS is a feature rather than the main purpose of the device. Those devices are the majority and may use [assisted GPS](http://en.wikipedia.org/wiki/Assisted_GPS) or standalone (not network dependent) or both. The vulnerability of consumer GPS to radio frequency interference from [planned wireless data services](http://en.wikipedia.org/wiki/LightSquared#Interference_issues) is controversial

**1.3 Uses of GPS System**

### 1.3.1 Dedicated GPS navigation devices

[](http://en.wikipedia.org/wiki/File:GPS_Receivers_2007.jpg)

Fig 1.3.1 Hand-held receivers

It’s generally wise to be prepared when heading into unfamiliar territory, and a GPS can provide protection if conditions turn out to be less than ideal.  Fog, rain, or snow can all be problematic and make a trail finding task that would be easy in good conditions, turn into a situation where one can easily get completely lost.

[](http://en.wikipedia.org/wiki/File:KyotoTaxiRide.jpg)

Fig 1.3.2 A [taxi](http://en.wikipedia.org/wiki/Taxicab) equipped with GPS

Dedicated devices have various degrees of mobility. Hand-held, outdoor, or sport receivers have replaceable batteries that can run them for several hours, making them suitable for [hiking](http://en.wikipedia.org/wiki/Hiking), [bicycle touring](http://en.wikipedia.org/wiki/Bicycle_touring) and other activities far from an electric power source. Their screens are small, and some do not show color, in part to save power. Cases are rugged and some are water resistant.

Other receivers, often called mobile are intended primarily for use in a car, but have a small rechargeable internal battery that can power them for an hour or two away from the car. Special purpose devices for use in a car may be permanently installed and depend entirely on the automotive electrical system.

The pre-installed embedded software of early receivers did not display maps; 21st century ones commonly show interactive street maps (of certain regions) that may also show [points of interest](http://en.wikipedia.org/wiki/Point_of_interest), route information and step-by-step routing directions, often in spoken form with a feature called “[text to speech](http://en.wikipedia.org/wiki/Text_to_speech)”.

### 1.3.2 Mobile phones with GPS capability

Due in part to regulations encouraging [mobile phone tracking](http://en.wikipedia.org/wiki/Mobile_phone_tracking), including [E911](http://en.wikipedia.org/wiki/E911), the majority of GPS receivers are built into [mobile telephones](http://en.wikipedia.org/wiki/Mobile_telephone), with varying degrees of coverage and user accessibility. Commercial navigation software is available for most 21st-century [smart phones](http://en.wikipedia.org/wiki/Smartphone) as well as some [Java](http://en.wikipedia.org/wiki/Java_%28programming_language%29)-enabled phones that allows them to use an internal or external GPS receiver (in the latter case, connecting via [serial](http://en.wikipedia.org/wiki/Serial_communications) or [Bluetooth](http://en.wikipedia.org/wiki/Bluetooth)). Some phones using [assisted GPS](http://en.wikipedia.org/wiki/Assisted_GPS) (A-GPS) function poorly when out of range of their carrier’s cell towers. Others can navigate worldwide with satellite GPS signals as well as a dedicated portable GPS receiver does, upgrading their operation to A-GPS mode when in range. Still others have a [hybrid positioning system](http://en.wikipedia.org/wiki/Hybrid_positioning_system) that can use other signals when GPS signals are inadequate.

More [bespoke](http://en.wikipedia.org/wiki/Bespoke) solutions also exist for [smart phones](http://en.wikipedia.org/wiki/Smartphone) with inbuilt GPS capabilities. Some such phones can use [tethering](http://en.wikipedia.org/wiki/Tethering) to double as a [wireless modem](http://en.wikipedia.org/wiki/Wireless_modem) for a [laptop](http://en.wikipedia.org/wiki/Laptop), while allowing GPS-navigation/ localization as well. One such example is marketed by [Verizon Wireless](http://en.wikipedia.org/wiki/Verizon_Wireless) in the [United States](http://en.wikipedia.org/wiki/United_States), and is called [VZ Navigator](http://en.wikipedia.org/wiki/VZ_Navigator). The system uses [gps one](http://en.wikipedia.org/wiki/GpsOne) technology to determine the location, and then uses the mobile phone’s data connection to download maps and calculate navigational routes. Other products including [iPhone](http://en.wikipedia.org/wiki/IPhone) are used to provide similar services. [Nokia](http://en.wikipedia.org/wiki/Nokia) gives [Ovi Maps](http://en.wikipedia.org/wiki/Ovi_Maps) free on its smart phones and maps can be preloaded.

According to market research from the independent analyst firm Berg Insight, the sales of GPS-enabled GSM/WCDMA handsets was 150 million units in 2009, while only 40 million separate GPS receivers were sold.

GPS navigation applications for mobile phones include on-line (e.g. [Waze](http://en.wikipedia.org/wiki/Waze), [Google Maps Navigation](http://en.wikipedia.org/wiki/Google_Maps_Navigation)) and off-line (e.g. [iGo](http://en.wikipedia.org/wiki/IGO_%28software%29) for Android, Maverick) navigation applications. [Google Maps Navigation](http://en.wikipedia.org/wiki/Google_Maps_Navigation), which is included with [Android](http://en.wikipedia.org/wiki/Android_%28operating_system%29), means most smart phone users only need their phone to have a [personal navigation assistant](http://en.wikipedia.org/wiki/Personal_navigation_assistant).

Many Android smart phones have an additional GPS feature, called **EPO** ([Extended Prediction Orbit](http://en.wikipedia.org/w/index.php?title=Extended_Prediction_Orbit&action=edit&redlink=1)). The phone downloads a file to help it locate GPS satellites more quickly and reduce the Time to First Fix.

### 1.3.3 Laptop PC GPS

Software companies have made available [GPS navigation software](http://en.wikipedia.org/wiki/GPS_navigation_software) programs for in-vehicle use on laptop computers Benefits of GPS on a laptop include larger map overview, ability to use the keyboard to control GPS functions, and some GPS software for laptops offers advanced trip-planning features not available on other platforms. Inbuilt GPS module is used to locate the position with proper latitude and longitude degree and software module prints the values on console.

### GPS modules

[](http://en.wikipedia.org/wiki/File:GPS_receiver_(mouse).jpg)

Fig 1.3.3 A modern SiRFstarIII chip based 20-channel GPS receiver with WAAS/EGNOS support

Other GPS devices need to be connected to a computer in order to work. This computer can be a [home computer](http://en.wikipedia.org/wiki/Home_computer), [laptop](http://en.wikipedia.org/wiki/Laptop), [PDA](http://en.wikipedia.org/wiki/Personal_digital_assistant), [digital camera](http://en.wikipedia.org/wiki/Digital_camera), or [smart phones](http://en.wikipedia.org/wiki/Smartphone). Depending on the type of computer and available connectors, connections can be made through a [serial](http://en.wikipedia.org/wiki/Serial_cable) or [USB](http://en.wikipedia.org/wiki/Universal_Serial_Bus) cable, as well as [Bluetooth](http://en.wikipedia.org/wiki/Bluetooth), [Compact Flash](http://en.wikipedia.org/wiki/CompactFlash), [SD](http://en.wikipedia.org/wiki/Secure_Digital_card), [PCMCIA](http://en.wikipedia.org/wiki/PC_Card) and the newer [Express Card](http://en.wikipedia.org/wiki/ExpressCard). Some PCMCIA/Express Card GPS units also include a [wireless modem](http://en.wikipedia.org/wiki/Wireless_modem).

Devices usually do not come with pre-installed [GPS navigation software](http://en.wikipedia.org/wiki/GPS_navigation_software), thus, once purchased; the user must install or write their own software. As the user can choose which software to use, it can be better matched to their personal taste. It is very common for a PC-based GPS receiver to come bundled with a navigation software suite. Also, GPS modules are significantly cheaper than complete stand-alone systems (around [€](http://en.wikipedia.org/wiki/Euro)50 to €100). The software may include maps only for a particular region, or the entire world, if software such as Google Maps, [Networks in Motion’s Atlas Book mobile navigation platform](http://en.wikipedia.org/wiki/VZ_Navigator), etc., are used.

Some hobbyists have also made some GPS devices and open-sourced the plans. Examples include the [Elektor GPS units](http://www.elektor.com/magazines/2008/october/multi-purpose-gps-receiver.684158.lynkx). These are based around a [SiRFstarIII](http://en.wikipedia.org/wiki/SiRFstarIII) chip and are comparable to their commercial counterparts. Other chips are also available.

## 1.3.4 Commercial aviation

Commercial aviation applications include GPS devices that calculate location and feed that information to large multi-input navigational computers for [autopilot](http://en.wikipedia.org/wiki/Autopilot), course information and correction displays to the pilots, and course tracking and recording devices.

## 1.3.5 Military

Military applications include devices similar to consumer sport products for foot soldiers (commanders and regular soldiers), small vehicles and ships, and devices similar to commercial aviation applications for aircraft and missiles. Examples are the United States military’s Commander’s Digital Assistant and the Soldier Digital Assistant. Prior to May 2000 only the military had access to the full accuracy of GPS. Consumer devices were restricted by [selective availability](http://en.wikipedia.org/wiki/Error_analysis_for_the_Global_Positioning_System#Selective_availability) (SA), which was scheduled to be phased out but was removed abruptly by President Clinton. [Differential GPS](http://en.wikipedia.org/wiki/Differential_GPS) is a method of cancelling out the error of SA and improving GPS accuracy, and has been routinely available in commercial applications such as for golf carts. GPS is limited to about 15 meter accuracy even without SA. DGPS can be within a few centimeters.

### 1.3.6 Advertisement

Since GPS devices can give the user’s exact location, this helps advertising agents to give more relevant advertisement to the users based on their current location. The agencies might promote shops which are nearby to the users, rather than totally irrelevant shops. The advertising agency also will store the user’s location for the agency’s future uses. However, the regulatory agents all around the world (especially USA and Europe) start to consider whether geo-location data should be a sensitive data or not. If the data is sensitive data, the marketing team of an agency cannot store geo-location of people since this a privacy violation.However, if the regulatory agents choose to consider geo-location as non-sensitive data, then private companies can have permission to store the user’s location in their database.

### 1.3.7 Surveillance

Privacy concerns also arise when employers use [GPS tracking units](http://en.wikipedia.org/wiki/GPS_tracking_unit) to track their employees’ location, for example using [vehicle tracking systems](http://en.wikipedia.org/wiki/Vehicle_tracking_system). This raises a major question about whether this violates personal privacy of employees. It raises a lot more concern for privacy violation if the employers collect geo-location data of their employee after work hours and during their holidays. In 2010, [New York Civil Liberties Union](http://en.wikipedia.org/wiki/New_York_Civil_Liberties_Union) filed a case against the Labor Department for firing Michael Cunningham after tracking Michael Cunningham’s daily activity and locations using GPS device that has attached in his car. This raises few questions regarding the limit of surveillance. The worst privacy violation is done by FBI when they tracked down Antoine Jones GPS devices even without any search warrants. Later the Federal Appeal Court rejected FBI’s surveillance data as a proof against Antoine Jones.

### 1.3.8 Stalking

GPS devices are also used by private investigators in order to give more information to their clients. They will plant their own GPS devices in order to know more about their target. Moreover, some rental car services use the same technique to prevent their customers from going out of their targeted area. They charge additional fees for those who violate their rules. They get this information by using the car’s GPS devices.

## Types of GPS trackers

Usually, a GPS tracker will fall into one of these three categories, though most [smart phones](http://en.wikipedia.org/wiki/Smartphone), being [GPS Phones](http://en.wikipedia.org/wiki/GPS_Phone), can work in all these modes, depending on which [mobile applications](http://en.wikipedia.org/wiki/Mobile_application) are installed:

A **GPS tracking** unit is a device that uses the [Global Positioning System](http://en.wikipedia.org/wiki/Global_Positioning_System) to determine the precise location of a vehicle, person, or other asset to which it is attached and to record the position of the asset at regular intervals. The recorded location data can be stored within the tracking unit, or it may be transmitted to a central location data base, or internet-connected computer, using a[cellular](http://en.wikipedia.org/wiki/Cellular_network) ([GPRS](http://en.wikipedia.org/wiki/GPRS) or [SMS](http://en.wikipedia.org/wiki/SMS)), [radio](http://en.wikipedia.org/wiki/Radio), or [satellite modem](http://en.wikipedia.org/wiki/Satellite_modem) embedded in the unit. This allows the asset’s location to be displayed against a map backdrop either in real time or when nalyzing the track later, using **GPS tracking software**. Data tracking software is available for smartphones with GPS capability

### Data loggers

[](http://en.wikipedia.org/wiki/File:Royaltek_rgm-3800_IMGP9822_OSM.jpg)

Fig 1.3.4 Typical GPS logger

A GPS logger simply [logs](http://en.wikipedia.org/wiki/Data_logging) the position of the device at regular intervals in its internal memory. Modern GPS loggers have either a [memory card](http://en.wikipedia.org/wiki/Memory_card) slot, or internal [flash memory](http://en.wikipedia.org/wiki/Flash_memory) and a [USB](http://en.wikipedia.org/wiki/Universal_Serial_Bus) port. Some act as a [USB flash drive](http://en.wikipedia.org/wiki/USB_flash_drive). This allows [downloading](http://en.wikipedia.org/wiki/Download) of the track log data for further analyzing in a computer. The track list or [point of interest](http://en.wikipedia.org/wiki/Point_of_interest) list may be in [GPX](http://en.wikipedia.org/wiki/GPS_eXchange_Format), [KML](http://en.wikipedia.org/wiki/Keyhole_Markup_Language), [NMEA](http://en.wikipedia.org/wiki/National_Marine_Electronics_Association) or other format.

Most [digital cameras](http://en.wikipedia.org/wiki/Digital_camera) save the time a photo was taken. Provided the camera clock was reasonably accurate or used GPS as its time source, this time can be correlated with GPS log data, to provide an accurate location. This can be added to the [Exif](http://en.wikipedia.org/wiki/Exif) metadata in the picture file. Cameras with GPS receiver built in can directly produce such a [geotagged photograph](http://en.wikipedia.org/wiki/Geotagged_photograph).

In some [Private Investigation](http://en.wikipedia.org/wiki/Private_Investigation) cases, data loggers are used to keep track of a target vehicle. The PI need not follow the target so closely, and always has a backup source of data.

### 1.4.1 Data pushers

Data pusher is the most common type of GPS tracking unit, used for [asset tracking](http://en.wikipedia.org/wiki/Asset_tracking), [personal tracking](http://en.wikipedia.org/w/index.php?title=Personal_tracking&action=edit&redlink=1) and [Vehicle tracking system](http://en.wikipedia.org/wiki/Vehicle_tracking).

Also known as a *GPS beacon*, this kind of device [pushes](http://en.wikipedia.org/wiki/Push_technology) (i.e. “sends”) the position of the device as well as other information like [speed](http://en.wikipedia.org/wiki/Speed) or [altitude](http://en.wikipedia.org/wiki/Altitude) at regular intervals, to a determined [server](http://en.wikipedia.org/wiki/Server_%28computing%29), that can store and instantly analyze the data.

A [GPS navigation device](http://en.wikipedia.org/wiki/GPS_navigation_device) and a mobile phone sit side-by-side in the same box, powered by the same battery. At regular intervals, the phone sends a text message via SMS or GPRS, containing the data from the GPS receiver. Newer GPS-integrated [smart phones](http://en.wikipedia.org/wiki/Smartphone) running GPS tracking software can turn the phone into a data pusher (or logger) device; as of 2009 [open source](http://en.wikipedia.org/wiki/Open_source) and [proprietary](http://en.wikipedia.org/wiki/Proprietary_software) applications are available for common [Java ME](http://en.wikipedia.org/wiki/Java_ME) enabled phones, [iPhone](http://en.wikipedia.org/wiki/IPhone), [Android](http://en.wikipedia.org/wiki/Android_%28operating_system%29), Windows Mobile, and Symbian.

Most 21st-century GPS trackers provide data “push” technology, enabling sophisticated GPS tracking in business environments, specifically organizations that employ a mobile workforce, such as a commercial fleet. Typical GPS tracking systems used in commercial [fleet management](http://en.wikipedia.org/wiki/Fleet_management) have two core parts: location hardware (or tracking device) and tracking software.This combination is often referred to as an [Automatic Vehicle Location](http://en.wikipedia.org/wiki/Automatic_Vehicle_Location) system. The tracking device is most often hardwiring installed in the vehicle; connected to the [CAN-bus](http://en.wikipedia.org/wiki/CAN-bus), [Ignition system](http://en.wikipedia.org/wiki/Ignition_system) switch, battery. It allows collection of extra data, which later gets transferred to the [GPS tracking server](http://en.wikipedia.org/wiki/GPS_tracking_server), where it is available for viewing, in most cases via a website accessed over the [internet](http://en.wikipedia.org/wiki/Internet), where fleet activity can be viewed live or historically using digital maps and reports.

GPS tracking systems used in commercial fleets are often configured to transmit location and telemetry input data at a set update rate or when an event (door open/close, auxiliary equipment on/off, [geofence](http://en.wikipedia.org/wiki/Geofence) border cross) triggers the unit to transmit data. Live GPS Tracking used in commercial fleets, generally refers to systems which update regularly at 1 minute, 2 minute or 5 minute intervals, whilst the ignition status is on. Some tracking systems combine timed updates with heading change triggered updates.

GPS tracking solutions are recently being used in mainstream commercial auto insurance these are sometimes called [Telematics 2.0](http://en.wikipedia.org/wiki/Telematics_2.0).

**1.4.2 The applications of these kinds of trackers include:**

#### Personal tracking

* [Law enforcement](http://en.wikipedia.org/wiki/Law_enforcement). An arrested suspect out on bail may have to wear a GPS tracker, usually an [ankle monitor](http://en.wikipedia.org/wiki/Ankle_monitor), as a bail condition.
* Race control. In some sports, such as [gliding](http://en.wikipedia.org/wiki/Gliding), participants are required to carry a tracker. This allows, among other applications, for race officials to know if the participants are cheating, taking unexpected shortcuts or how far apart they are. This use has been featured in the movie [*Rat Race*](http://en.wikipedia.org/wiki/Rat_Race_%28film%29).
* [Espionage](http://en.wikipedia.org/wiki/Espionage)/[surveillance](http://en.wikipedia.org/wiki/Surveillance). When put on a person, or on his personal vehicle, it allows the person monitoring the tracking to know his/her habits. This application is used by [private investigators](http://en.wikipedia.org/wiki/Private_investigator).
* These devices are also used by some parents to track their children. The supporters claim that if cleverly used, this actually allows children more independence.
* GPS personal tracking devices assist in the care of the elderly and vulnerable. Devices allow users to call for assistance and optionally allow designated careers to locate the user’s position, typically within 5 to 10 meters. Their use helps promote independent living and social inclusion for the elderly. Devices often incorporate either 1-way or 2-way voice communication which is activated by pressing a button or sliding a switch. Some devices also allow the user to call several phone numbers using pre-programmed speed dial buttons. Trials using GPS personal tracking devices are also underway in several countries for use with early stage dementia.
* [Internet](http://en.wikipedia.org/wiki/Internet) Fun. Some [Web 2.0](http://en.wikipedia.org/wiki/Web_2.0) pioneers have created their own personal web pages that show their position constantly, and in real-time, on a map within their website. These usually use data push from a GPS enabled cell phone or a personal GPS tracker.
* Adventure sports. GPS tracking devices such as the [SPOT Satellite Messenger](http://en.wikipedia.org/wiki/SPOT_Satellite_Messenger) are available that allow friends, family, and rescue personnel to track the position of a person and receive messages from them, even when the user of the device is outside of cellular range.
* Monitoring the employees. GPS handled tracking devices with built in cell phone are used to monitor the employees by various companies, specially the employees which are engaged in field work job.

#### Asset tracking

* Solar Powered. The advantage of some solar powered units is that they have much more power over their lifetime than battery powered units. This gives them the advantage to report their position and status much more often than battery units which need to conserve their energy to extend their life. Some wireless solar powered units, such as the [Rail Rider](http://en.wikipedia.org/wiki/RailRider) can report more than 20,000 times per year and work indefinitely on solar power eliminating the need to change batteries.
* Animal control. When put on a wildlife animal (e.g. in a collar), it allows scientists to study its activities and migration patterns. Vaginal implant transmitters mark the location where pregnant females give birth. [Animal tracking collars](http://en.wikipedia.org/wiki/Tagg) may also be put on domestic animals, to locate them in case they get lost.

### *Data pullers*

GPS data pullers are also known as *GPS transponders*. Contrary to data pushers, that send the position of the devices at regular intervals ([push technology](http://en.wikipedia.org/wiki/Push_technology)), these devices are always-on and can be queried as often as required ([pull technology](http://en.wikipedia.org/wiki/Pull_technology)). This technology is not in widespread use, but an example of this kind of device is a computer connected to the [Internet](http://en.wikipedia.org/wiki/Internet) and running [gpsd](http://en.wikipedia.org/wiki/Gpsd).

These can often be used in the case where the location of the tracker will only need to be known occasionally e.g. placed in property that may be stolen, or that does not have constant source of energy to send data on a regular basis, like freights or containers.

Data Pullers are coming into more common usage in the form of devices containing a GPS receiver and a cell phone which, when sent a special [SMS](http://en.wikipedia.org/wiki/SMS) message reply to the message with their location.

## 

## *Uses in Marketing*

In August, 2010, Brazilian company Unilever ran a promotion where GPS trackers were placed in boxes of Omo laundry detergent. Teams would then track consumers who purchased the boxes of detergent to their homes where they would be awarded with a prize for their purchase. The company also launched a website (in Portuguese) to show the approximate location of the winners’ homes. The manner in which GPS data is collected today can be highly fragmented, with large missing values and high variability in the frequency and spread of the location points.

**CHAPTER 2**

**2.GPS MODULE WITH ARDUINO**

**2.1 Overview of GPS Module**

The first thing we will do is test the GPS by connecting it directly to the computer. This is useful because sometimes you may want to get geo locative data directly into software. You don’t need to do this step, but it can be helpful to determine if your GPS module is working - one less thing to worry about later.

**2.2 GPS with Arduino**

You can connect with an FTDI cable or use your Arduino

**2.2.1 Wiring using an FTDI cable**

The easiest way to connect the GPS module to a computer is to remove it from the Arduino and connect the GPSPWR line to ground.

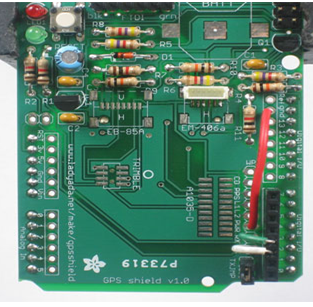
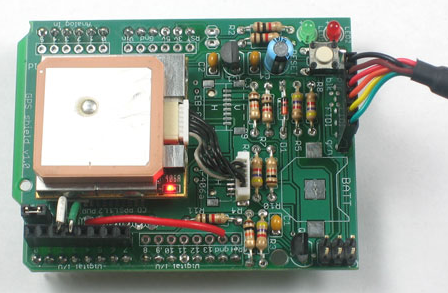


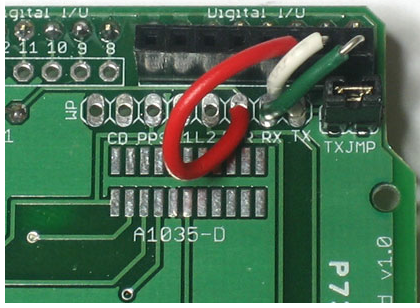
Fig 2.2.1 Wiring using an FTDI cable

Then plug in an FTDI cable and connect using a serial terminal program (or the Arduino IDE, see below)



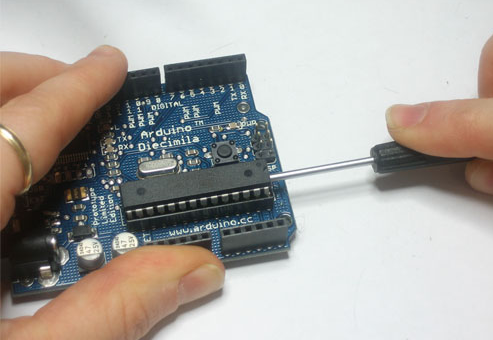
**2.2.2 Wiring using an Arduino**

Remove the shield from the Arduino. Connect the GPS TX line to digital pin 1 and the RXline to digital pin 0. Connect the PWR line to digital pin 2.

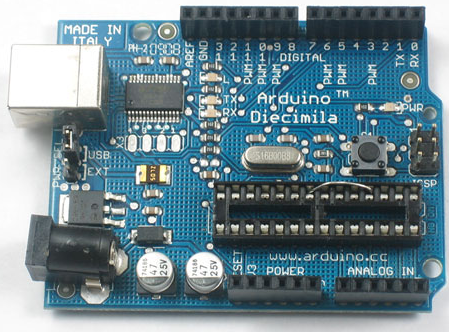


**Remove chip**

We want the GPS unit to talk to the FT232 chip on the arduino which will let us listen in using USB, but the problem is that the Arduino chip (ATmega168) is in the way so we must remove it. First, gentlypry the Arduino microcontroller from its socket using a small flat screwdriver or similar. Try to make sure the pins don’t get bent. Put it in a safe place. Preferably in an anti-static bag

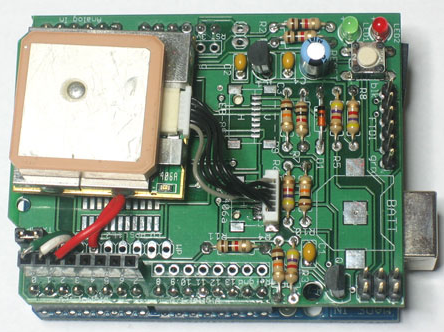


Next we will jumper digital i/o pin 2 to ground (LOW) which will make sure the GPS unit is turned on when we connect up. Use a spare piece of wire and plug them into the empty socket as shown. Triple check to make sure you have the jumper in the proper socket holes!

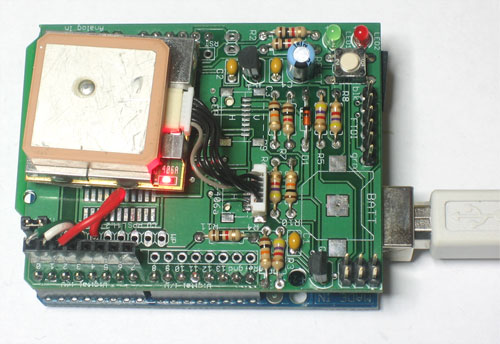


**Plug in & power up**

Plug the shield into the Arduino, and plug the GPS module into the little connector



Now connect the Arduino to your computer via USB. The GPS module should light up, indicating that it’s on. If the GPS module doesn’t turn on, check the PWR jumper is connected to digital pin 2, that the socket jumper is correct, and that the Arduino is powered.

[](http://www.ladyada.net/images/gpsshield/directconnected.jpg)

**CHAPTER 3**

**3.HARDWARE COMPONENTS**

**3.1 ATMega328P and features**

ATMega328 is the high-performance, low-power Atmel 8-bit AVR RISC-based microcontroller combines 32KB of programmable flash memory, 2KB SRAM, 1KB EEPROM, an 8-channel 10-bit A/D converter, and a JTAG interface for on-chip debugging. The device supports throughput of 16 MIPS at 16 MHz and operates between 4.5-5.5 volts.

By executing instructions in a single clock cycle, the device achieves throughputs approaching 1 MIPS per MHz, balancing power consumption and processing speed.

**3.2 Pin Description**

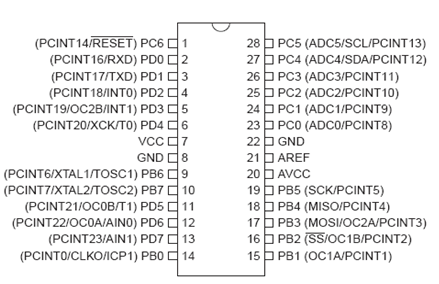


Fig 3.2.1 ATMega328 pin diagram

**VCC**

Digital supply voltage

**GND**

Ground

**Port B (PB7:0) XTAL1/XTAL2/TOSC1/TOSC2**

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The

Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pi ns that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running. Depending on the clock selection fuse settings, PB6 can be used as input to the inverting Oscillator amplifier and input to the internal clock operating circuit. Depending on the clock selection fuse settings, PB7 can be used as output from the inverting Oscillator amplifier. If the Internal Calibrated RC Oscillator is used as chip clock source, PB7..6 is used as TOSC2..1 input for the Asynchronous Timer/Counter2 if the AS2 bit in ASSR is set.

**Port C (PC5:0)**

Port C is a 7-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The PC5..0 output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running.

**PC6/RESET**

If the RSTDISBL Fuse is programmed, PC6 is used as an I/O pin. Note that the electrical characteristics of PC6 differ from those of the other pins of Port C. If the RSTDISBL Fuse is unprogrammed.PC6 is used as a Reset input. A low level on this pin for longer than the minimum pulse length will generate a Reset, even if the clock is not running.

**Port D (PD7:0)**

Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running.

**AVCC**

AVCC is the supply voltage pin for the A/D Converter, PC3:0, and ADC7:6. It should be externally connected to VCC, even if the ADC is not used. If the ADC is used, it should be connected to VCC through a low-pass filter. Note that PC6..4 use digital supply voltage, VCC

**AREF**

AREF is the analog reference pin for the A/D Converter.

**3.2.1 Block Diagram**

The ATmega48PA/88PA/168PA/328P is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega328P achieves throughputs approaching 1 MIPS per MHz allowing the system designed to optimize power consumption versus processing speed.

* Advanced RISC Architecture
* 131 Powerful Instructions – Most Single Clock Cycle Execution
* 32 x 8 General Purpose Working Registers
* Fully Static Operation
* Up to 20 MIPS Throughput at 20 MHz
* On-chip 2-cycle Multiplier
* High Endurance Non-volatile Memory Segments
* 4/8/16/32K Bytes of In-System Self-Programmable Flash progam memory
* (ATmega48PA/88PA/168PA/328P)
* 256/512/512/1K Bytes EEPROM (ATmega48PA/88PA/168PA/328P)
* 512/1K/1K/2K Bytes Internal SRAM (ATmega48PA/88PA/168PA/328P)
* Write/Erase Cycles: 10,000 Flash/100,000 EEPROM
* Data retention: 20 years at 85°C/100 years at 25°C(1)
* Optional Boot Code Section with Independent Lock Bits
* In-System Programming by On-chip Boot Program
* Programming Lock for Software Security Peripheral Features
* Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
* One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
* Real Time Counter with Separate Oscillator
* Six PWM Channels
* 8-channel 10-bit ADC in TQFP and QFN/MLF package
* 6-channel 10-bit ADC in PDIP Package
* Programmable Serial USART
* Master/Slave SPI Serial Interface
* Byte-oriented 2-wire Serial Interface (Philips I2C compatible)
* Programmable Watchdog Timer with Separate On-chip Oscillator
* On-chip Analog Comparator
* Interrupt and Wake-up on Pin Change
* Special Microcontroller Features
* Power-on Reset and Programmable Brown-out Detection
* Internal Calibrated Oscillator
* External and Internal Interrupt Sources
* Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby, and Extended Standby
* I/O and Packages
* 23 Programmable I/O Lines
* 28-pin PDIP, 32-lead TQFP, 28-pad QFN/MLF and 32-pad QFN/MLF
* Operating Voltage: 1.8 - 5.5V for ATmega48PA/88PA/168PA/328P
* Temperature Range: -40°C to 85°C
* Speed Grade: 0 - 20 MHz @ 1.8 - 5.5V
* Low Power Consumption at 1 MHz, 1.8V, 25°C for ATmega48PA/88PA/168PA/328P:
* Active Mode: 0.2 mA
* Power-down Mode: 0.1 µA

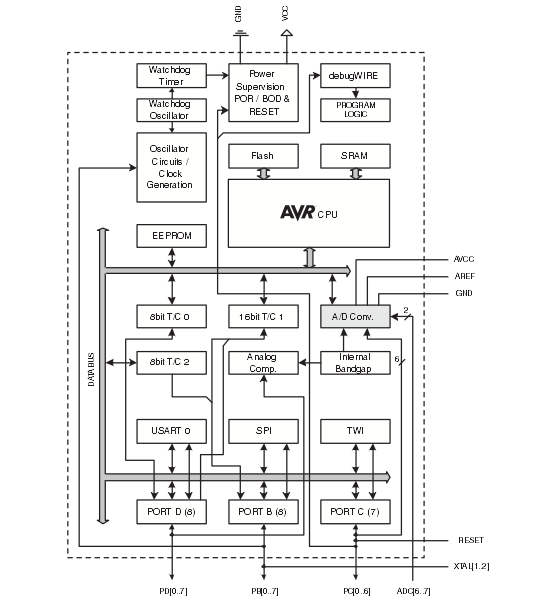


Fig 3.3.2 ATmega48PA/88PA/168PA/328P is a low-power CMOS 8-bit microcontroller

The AVR core combines a rich instruction set with 32 general purpose working registers. All the

32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

The ATmega328P provides the following features: 8K bytes of In-System Programmable Flash with Read-While-Write capabilities,1K bytes EEPROM, 2K bytes SRAM, 23 general purpose I/O lines, 32 general purpose working registers, three flexible Timer/Counters with compare modes, internal and external interrupts, a serial programmable USART, a byte-oriented 2-wire Serial Interface, an SPI serial port, a 6-channel 10-bit ADC (8 channels in TQFP and QFN/MLF packages), a programmable Watchdog Timer with internal Oscillator, and five software selectable power saving modes. The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, USART, 2-wire Serial Interface, SPI port, and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next interrupt or hardware reset. In Power-save mode, the asynchronous timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except asynchronous timer and ADC, to minimize switching noise during ADC conversions. In Standby mode, the crystal/resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low power consumption. The device is manufactured using Atmel’s high density non-volatile memory technology. The On-chip ISP Flash allows the program memory to be reprogrammed In-System through an SPI serial interface, by a conventional non-volatile memory programmer, or by an On-chip Boot program running on the AVR core. The Boot program can use any interface to download the application program in the Application Flash memory. Software in the Boot Flash section will continue to run while the Application Flash section is updated, providing true Read-While-Write operation. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a monolithic chip, the Atmel ATmega328P is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications. The ATmega328P AVR is supported with a full suite of program and system development tools including: C Compilers, Macro Assemblers, and Program Debugger/Simulators, In-Circuit Emulators, and Evaluation kit.

**CHAPTER 4**

**4.SOFTWARE INTRODUCTION**

**4.1 Introduction**

Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. It's intended for artists, designers, hobbyists, and anyone interested in creating interactive objects or environments.

Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. The microcontroller on the board is programmed using the [Arduino programming language](http://arduino.cc/en/Reference/HomePage) (based on [Wiring](http://wiring.org.co/)) and the Arduino development environment (based on [Processing](http://www.processing.org/)). Arduino projects can be stand-alone or they can communicate with software running on a computer (e.g. Flash, Processing, MaxMSP).

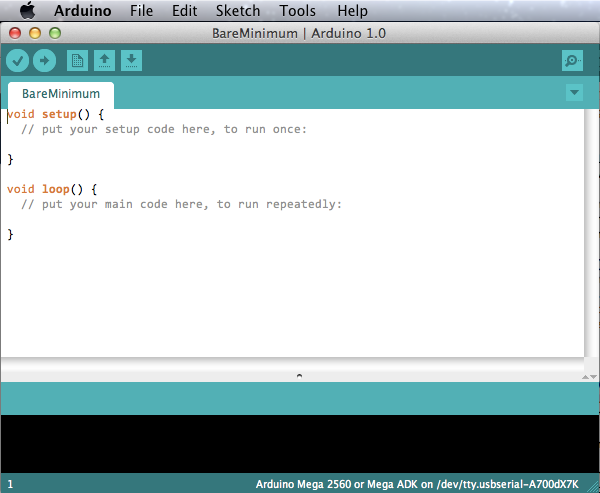
The boards can be [built by hand](http://arduino.cc/en/Main/ArduinoBoardSerialSingleSided3) or [purchased](http://arduino.cc/en/Main/Buy) preassembled; the software can be [downloaded](http://arduino.cc/en/Main/Software) for free. The hardware reference designs (CAD files) are [available](http://arduino.cc/en/Main/Products) under an open-source license, you are free to [adapt them to your needs](http://arduino.cc/en/Main/Policy).

Arduino received an Honorary Mention in the Digital Communities section of the 2006 Ars Electronica Prix. The Arduino team is: [Massimo Banzi](http://www.tinker.it/), [David Cuartielles](http://www.blushingboy.org/), [Tom Igoe](http://tigoe.net/), [Gianluca Martino](http://www.smartprojects.it/), and [David Mellis](http://dam.mellis.org/).

The Arduino [integrated development environment](http://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) is a [cross-platform](http://en.wikipedia.org/wiki/Cross-platform) application written in [Java](http://en.wikipedia.org/wiki/Java_(programming_language)), and is derived from the IDE for the [Processing programming language](http://en.wikipedia.org/wiki/Processing_(programming_language)) and the [Wiring](http://en.wikipedia.org/wiki/Wiring_(development_platform)) projects. It is designed to introduce programming to artists and other newcomers unfamiliar with software development. It includes a code editor with features such as [syntax highlighting](http://en.wikipedia.org/wiki/Syntax_highlighting), [brace matching](http://en.wikipedia.org/wiki/Brace_matching), and automatic indentation, and is also capable of compiling and uploading programs to the board with a single click. A program or code written for Arduino is called a "sketch".

**4.2 Software**

**What is Arduino?**

s

Arduino is a tool for making computers that can sense and control more of the physical world than your desktop computer. It's an open-source physical computing platform based on a simple microcontroller board, and a development environment for writing software for the board.

Arduino can be used to develop interactive objects, taking inputs from a variety of switches or sensors, and controlling a variety of lights, motors, and other physical outputs. Arduino projects can be stand-alone, or they can be communicating with software running on your computer (e.g. Flash, Processing, MaxMSP.) The boards can be assembled by hand or purchased preassembled; the open-source IDE can be downloaded for free.

The Arduino programming language is an implementation of Wiring, a similar physical computing platform, which is based on the Processing multimedia programming environment.

**Why Arduino?**

There are many other microcontrollers and microcontroller platforms available for physical computing. Parallax Basic Stamp, Netmedia's BX-24, Phidgets, MIT's Handyboard, and many others offer similar functionality. All of these tools take the messy details of microcontroller programming and wrap it up in an easy-to-use package. Arduino also simplifies the process of working with microcontrollers, but it offers some advantage for teachers, students, and interested amateurs over other systems:

* **Inexpensive** - Arduino boards are relatively inexpensive compared to other microcontroller platforms. The least expensive version of the Arduino module can be assembled by hand, and even the pre-assembled Arduino modules cost less than $50
* **Cross-platform** - The Arduino software runs on Windows, Macintosh OSX, and Linux operating systems. Most microcontroller systems are limited to Windows.
* **Simple, clear** **programming environment** - The Arduino programming environment is easy-to-use for beginners, yet flexible enough for advanced users to take advantage of as well. For teachers, it's conveniently based on the Processing programming environment, so students learning to program in that environment will be familiar with the look and feel of Arduino
* **Open source and extensible software**- The Arduino software is published as open source tools, available for extension by experienced programmers. The language can be expanded through C++ libraries, and people wanting to understand the technical details can make the leap from Arduino to the AVR C programming language on which it's based. Similarly, you can add AVR-C code directly into your Arduino programs if you want to.
* **Open source and extensible hardware** - The Arduino is based on Atmel's ATMEGA8 and ATMEGA168microcontrollers. The plans for the modules are published under a Creative Commons license, so experienced circuit designers can make their own version of the module, extending it and improving it. Even relatively inexperienced users can build the breadboard version of the module in order to understand how it works and save money.

**4.3 Installation**

Step- by - step instructions for setting up the Arduino software and connecting it to an Arduino Uno, Mega2560, Duemilanove, Mega, or Diecimila

**Getting Started with Arduino on Windows**

**1 |** Get an Arduino board and USB cable

In this tutorial, we assume you're using an [Arduino Uno](http://arduino.cc/en/Main/ArduinoBoardUno), [Arduino Duemilanove](http://arduino.cc/en/Main/ArduinoBoardDuemilanove), [Nano](http://arduino.cc/en/Main/ArduinoBoardNano), or [Diecimila](http://arduino.cc/en/Main/ArduinoBoardDiecimila). If you have another board, read the corresponding page in this getting started guide.

You also need a standard USB cable (A plug to B plug): the kind you would connect to a USB printer, for example. (For the Arduino Nano, you'll need an A to Mini-B cable instead.)

**2 |** Download the Arduino environment

Get the latest version from the [download page](http://arduino.cc/en/Main/Software).

When the download finishes, unzip the downloaded file. Make sure to preserve the folder structure. Double-click the folder to open it. There should be a few files and sub-folders inside.

**3** | Connect the board

The Arduino Uno, Mega, Duemilanove and Arduino Nano automatically draw power from either the USB connection to the computer or an external power supply. If you're using an Arduino Diecimila, you'll need to make sure that the board is configured to draw power from the USB connection. The power source is selected with a jumper, a small piece of plastic that fits onto two of the three pins between the USB and power jacks. Check that it's on the two pins closest to the USB port.

Connect the Arduino board to your computer using the USB cable. The green power LED (labeled **PWR**) should go on.

**4** | Install the drivers

Installing drivers for the [Arduino Uno](http://arduino.cc/en/Main/ArduinoBoardUno) with Windows7, Vista, or XP:

* Plug in your board and wait for Windows to begin it's driver installation process. After a few moments, the process will fail, despite its best efforts
* Click on the Start Menu, and open up the Control Panel.
* While in the Control Panel, navigate to System and Security. Next, click on System. Once the System window is up, open the Device Manager.
* Look under Ports (COM & LPT). You should see an open port named "Arduino UNO (COMxx)"
* Right click on the "Arduino UNO (COmxx)" port and choose the "Update Driver Software" option.
* Next, choose the "Browse my computer for Driver software" option.
* Finally, navigate to and select the Uno's driver file, named **"ArduinoUNO.inf"**, located in the "Drivers" folder of the Arduino Software download (not the "FTDI USB Drivers" sub-directory).
* Windows will finish up the driver installation from there.

See also: [step-by-step screenshots for installing the Uno under Windows XP](http://arduino.cc/en/Guide/UnoDriversWindowsXP).

**Installing drivers for the** [**Arduino Duemilanove**](http://arduino.cc/en/Main/ArduinoBoardDuemilanove)**,** [**Nano**](http://arduino.cc/en/Main/ArduinoBoardNano)**, or** [**Diecimila**](http://arduino.cc/en/Main/ArduinoBoardDiecimila) **with Windows7, Vista, or XP:**

When you connect the board, Windows should initiate the driver installation process (if you haven't used the computer with an Arduino board before).

On Windows Vista, the driver should be automatically downloaded and installed. (Really, it works!)

On Windows XP, the Add New Hardware wizard will open:

* When asked **Can Windows connect to Windows Update to search for software?** Select **No, not this time**. Click next.
* Select **Install from a list or specified location (Advanced)** and click next.
* Make sure that **Search for the best driver in these locations** is checked; uncheck **Search removable media**; check **include this location in the search** and browse to the **drivers/FTDI USB Drivers** directory of the Arduino distribution. (The latest version of the drivers can be found on the [FTDI website](http://www.ftdichip.com/Drivers/VCP.htm).) Click next.
* The wizard will search for the driver and then tell you that a "USB Serial Converter" was found. Click finish.
* The new hardware wizard will appear again. Go through the same steps and select the same options and location to search. This time, a "USB Serial Port" will be found.

You can check that the drivers have been installed by opening the Windows Device Mananger (in the Hardware tab of System control panel). Look for a "USB Serial Port" in the Ports section; that's the Arduino board.

**5 |** Launch the Arduino application

Double-click the Arduino application. (Note: if the Arduino software loads in the wrong language, you can change it in the preferences dialog. See [the environment page](http://arduino.cc/en/Guide/Environment#languages) for details.)

**6** | Open the blink example

Open the LED blink example sketch: **File > Examples > 1.Basics > Blink**.

**7** | Select your board

You'll need to select the entry in the **Tools > Board** menu that corresponds to your Arduino.

For Duemilanove Arduino boards with an ATmega328 (check the text on the chip on the board), select **Arduino Duemilanove or Nano w/ ATmega328**. Previously, Arduino boards came with an ATmega168; for those, select**Arduino Diecimila, Duemilanove, or Nano w/ ATmega168**. (Details of the board menu entries are available [on the environment page](http://arduino.cc/en/Guide/Environment#boards).)

**8** | Select your serial port

Select the serial device of the Arduino board from the Tools | Serial Port menu. This is likely to be **COM3** or higher (**COM1** and **COM2** are usually reserved for hardware serial ports). To find out, you can disconnect your Arduino board and re-open the menu; the entry that disappears should be the Arduino board. Reconnect the board and select that serial port.

**9** | Upload the program

Now, simply click the "Upload" button in the environment. Wait a few seconds - you should see the RX and TX leds on the board flashing. If the upload is successful, the message "Done uploading." will appear in the status bar. (*Note:* If you have an Arduino Mini, NG, or other board, you'll need to physically present the reset button on the board immediately before pressing the upload button.)

A few seconds after the upload finishes, you should see the pin 13 (L) LED on the board start to blink (in orange). If it does, congratulations! You've gotten Arduino up-and-running.

If you have problems, please see the [troubleshooting suggestions](http://arduino.cc/en/Guide/Troubleshooting).

**4.4 Environment**

The Arduino development environment contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions, and a series of menus. It connects to the Arduino hardware to upload programs and communicate with them.

Writing Sketches

Software written using Arduino is called sketches. These sketches are written in the text editor. Sketches are saved with the file extension .ino. It has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino environment including complete error messages and other information. The bottom right hand corner of the window displays the current board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.

|  |  |
| --- | --- |
|  | *Verify*  Checks your code for errors. |
|  | *Upload*  Compiles your code and uploads it to the Arduino I/O board. See [uploading](http://arduino.cc/en/Guide/Environment#uploading) below for details.  Note: If you are using an external programmer, you can hold down the "shift" key on your computer when using this icon. The text will change to "Upload using Programmer" |
|  | *New*  Creates a new sketch. |
|  | *Open*  Presents a menu of all the sketches in your sketchbook. Clicking one will open it within the current window.  Note: due to a bug in Java, this menu doesn't scroll; if you need to open a sketch late in the list, use the **File | Sketchbook** menu instead. |
|  | *Save*  Saves your sketch. |
|  | *Serial Monitor*  Opens the [serial monitor](http://arduino.cc/en/Guide/Environment#serialmonitor). |

Additional commands are found within the five menus: **File**, **Edit**, **Sketch**, **Tools**, **Help**. The menus are context sensitive which means only those items relevant to the work currently being carried out are available.

Edit

* *Copy for Forum*

Copies the code of your sketch to the clipboard in a form suitable for posting to the forum, complete with syntax coloring

* *Copy as HTML*

Copies the code of your sketch to the clipboard as HTML, suitable for embedding in web pages

Sketch

* *Verify/Compile*

Checks your sketch for errors

* *Show Sketch Folder*

Opens the current sketch folder

* *Add File*

Adds a source file to the sketch (it will be copied from its current location). The new file appears in a new tab in the sketch window. Files can be removed from the sketch using the tab menu.

* *Import Library*

Adds a library to your sketch by inserting #include statements at the code of your code. For more details, see [libraries](http://arduino.cc/en/Guide/Environment#libraries) below

**Tool**

* *AutoFormat*

This formats your code nicely: i.e. indents it so that opening and closing curly braces line up, and that the statements inside curly braces are indented more

* *Archive Sketch*

Archives a copy of the current sketch in zip format. The archive is placed in the same directory as the sketch.

* *Board*

Select the board that you're using. See below for [descriptions of the various boards](http://arduino.cc/en/Guide/Environment#boards).

* *Serial Port*

This menu contains all the serial devices (real or virtual) on your machine. It should automatically refresh every time you open the top-level tools menu.

* *Programmer*

For selecting a hardware programmer when programming a board or chip and not using the onboard USB-serial connection. Normally you won't need this, but if you're [burning a boot loader](http://arduino.cc/en/Tutorial/Bootloader) to a new microcontroller, you will use this.

* *Burn Bootloader*

The items in this menu allow you to burn a [bootloader](http://arduino.cc/en/Tutorial/Bootloader) onto the microcontroller on an Arduino board. This is not required for normal use of an Arduino board but is useful if you purchase a new AT mega microcontroller (which normally comes without a bootloader). Ensure that you've selected the correct board from the Boards menu before burning the bootloader.

**Sketchbook**

The Arduino environment uses the concept of a sketchbook: a standard place to store your programs (or sketches). The sketches in your sketchbook can be opened from the File **>** Sketchbook menu or from the Open button on the toolbar. The first time you run the Arduino software, it will automatically create a directory for your sketchbook. You can view or change the location of the sketchbook location from with the Preferences dialog.

'''Beginning with version 1.0, files are saved with an .ino file extension. Previous versions use the .pde extension. You may still open .pde named files in version 1.0 and later, the software will automatically rename the extension to .ino.

**Tabs, Multiple Files, and Compilation**

Allows you to manage sketches with more than one file (each of which appears in its own tab). These can be normal Arduino code files (no extension), C files (.c extension), C++ files (.cpp), or header files (.h).

**Uploading**

Before uploading your sketch, you need to select the correct items from the Tools > Board and Tools > Serial Portmenus. The [boards](http://arduino.cc/en/Guide/Environment#boards) are described below. On the Mac, the serial port is probably something like/dev/tty.usbmodem241 (for an Uno or Mega2560 or Leonardo) or /dev/tty.usbserial-1B1 (for a Duemilanove or earlier USB board), or /dev/tty.USA19QW1b1P1.1 (for a serial board connected with a Keyspan USB-to-Serial adapter). On Windows, it's probably COM1 or COM2 (for a serial board) or COM4, COM5, COM7, or higher (for a USB board) - to find out, you look for USB serial device in the ports section of the Windows Device Manager. On Linux, it should be **/**dev/ttyUSB0, /dev/ttyUSB1 or similar.

Once you've selected the correct serial port and board, press the upload button in the toolbar or select the Upload item from the File menu. Current Arduino boards will reset automatically and begin the upload. With older boards (pre-Diecimila) that lack auto-reset, you'll need to press the reset button on the board just before starting the upload. On most boards, you'll see the RX and TX LEDs blink as the sketch is uploaded. The Arduino environment will display a message when the upload is complete, or show an error.

When you upload a sketch, you're using the Arduino bootloader, a small program that has been loaded on to the microcontroller on your board. It allows you to upload code without using any additional hardware. The bootloader is active for a few seconds when the board resets; then it starts whichever sketch was most recently uploaded to the microcontroller. The bootloader will blink the on-board (pin 13) LED when it starts (i.e. when the board resets).

**Libraries**

Libraries provide extra functionality for use in sketches, e.g. working with hardware or manipulating data. To use a library in a sketch, select it from the Sketch > Import Library menu. This will insert one or more #includes statements at the top of the sketch and compiles the library with your sketch. Because libraries are uploaded to the board with your sketch, they increase the amount of space it takes up. If a sketch no longer needs a library, simply delete its #include statements from the top of your code.

There is a [list of libraries](http://arduino.cc/en/Reference/Libraries) in the reference. Some libraries are included with the Arduino software. Others can be downloaded from a variety of sources. See these [instructions for installing a third-party library](http://arduino.cc/en/Guide/Libraries).

To write your own library, see [this tutorial](http://arduino.cc/en/Hacking/LibraryTutorial).

Third-Party Hardware

Support for third-party hardware can be added to the hardware directory of your sketchbook directory. Platforms installed there may include board definitions (which appear in the board menu), core libraries, bootloaders, and programmer definitions. To install, create the hardware directory, then unzip the third-party platform into its own sub-directory. (Don't use "arduino" as the sub-directory name or you'll override the built-in Arduino platform.) To uninstall, simply delete its directory.

For details on creating packages for third-party hardware, see the [platforms page](http://code.google.com/p/arduino/wiki/Platforms) on the Arduino Google Code developer’s site.

Serial Monitor

Displays serial data being sent from the Arduino board (USB or serial board). To send data to the board, enter text and click on the "send" button or press enter. Choose the baud rate from the drop-down that matches the rate passed to Serial. begin in your sketch. Note that on Mac or Linux, the Arduino board will reset (rerun your sketch from the beginning) when you connect with the serial monitor.

Installing Additional Arduino Libraries

What are Libraries?

Libraries are a collection of code that makes it easy for you to connect to a sensor, display, module, etc. For example, the built-in Liquid Crystal library makes it easy to talk to character LCD displays. There are hundreds of additional libraries available on the Internet for download. The built-in libraries and some of these additional libraries are [listed in the reference](http://arduino.cc/en/Reference/Libraries). To use the additional libraries, you will need to install them.

How to Install a Library

Libraries are often distributed as a ZIP file or folder. The name of the folder is the name of the library. Inside the folder will be a .cpp file, a .h file and often a keywords.txt file, examples folder, and other files required by the library. To install the library, first quit the Arduino application. Then uncompressed the ZIP file containing the library. For example, if you're installing a library called "Arduino Party", uncompressed Arduino Party.zip. It should contain a folder called Arduino Party, with files like ArduinoParty.cpp and Arduino Party.h inside. (If the .cpp and .h files aren't in a folder, you'll need to create one. In this case, you'd make a folder called "Arduino Party" and move into it all the files that were in the ZIP file, like ArduinoParty.cpp and Arduino Party.h.)Drag the Arduino Party folder into this folder (your libraries folder). Under Windows, it will likely be called "My Documents\Arduino\libraries". For Mac users, it will likely be called "Documents/Arduino/libraries". On Linux, it will be the "libraries" folder in your sketch book. Your Arduino library folder should now look like this (on Windows):

My Documents\Arduino\libraries\Arduino Party\ArduinoParty.cpp

My Documents\Arduino\libraries\Arduino Party\Arduino Party.h

My Documents\Arduino\libraries\Arduino Party\examples

...

or like this (on Mac):

Documents/Arduino/libraries/Arduino Party/ArduinoParty.cpp

Documents/Arduino/libraries/Arduino Party/ArduinoParty.h

Documents/Arduino/libraries/Arduino Party/examples

...

or similarly for Linux.

There may be more files than just the .cpp and .h files, just make sure they're all there.

(The library won't work if you put the .cpp and .h files directly into the libraries folder or if they're nested in an extra folder. For example:

Documents\Arduino\libraries\ArduinoParty.cpp and

Documents\Arduino\libraries\ArduinoParty\ArduinoParty\ArduinoParty.cpp

won't work.)

Restart the Arduino application. Make sure the new library appears in the Sketch->Import Library menu item of the software.

**4.5 Troubleshooting**

1. **Why I can't upload my programs to the Arduino board?**

There are many pieces involved in getting a program onto your Arduino board, and if any of them aren't right, the upload can fail. They include: the drivers for the board, the board and serial port selections in the Arduino software, access to the serial port, the physical connection to the board, the firmware on the 8U2 (on the Uno and Mega 2560), the bootloader on the main microcontroller on the board, the microcontroller's fuse settings, and more. Here are some specific suggestions for troubleshooting each of the pieces.

***Arduino Software***

* Make sure you have the right item selected in the **Tools > Board** menu. If you have an Arduino Uno, you'll need to choose it. Also, newer Arduino Duemilanove boards come with an ATmega328, while older ones have an ATmega168. To check, read the text on the microcontroller (the larger chip) on your Arduino board. For more information on the board menu items, see the [guide to the Arduino environment](http://arduino.cc/en/Guide/Environment).
* Then, check that the proper port is selected in the **Tools > Serial Port** menu (if your port doesn't appear, try restarting the IDE with the board connected to the computer). On the Mac, the serial port should be something like /dev/tty.usbmodem621 (for the Uno or Mega 2560) or /dev/tty.usbserial-A02f8e (for older, FTDI-based boards). On Linux, it should be /dev/ttyACM0 or similar (for the Uno or Mega 2560) or /dev/ttyUSB0 or similar (for older boards). On Windows, it will be a COM port but you'll need to check in the Device Manager (under Ports) to see which one. If you don't seem to have a serial port for your Arduino board, see the following information about drivers.

***Drivers***

Drivers provide a way for software on your computer (i.e. the Arduino software) to talk to hardware you connect to your computer (the Arduino board). In the case of Arduino, the drivers work by providing a virtual serial port (or virtual COM port). The Arduino Uno and Mega 2560 use standard drivers (USB CDC) provided by the operating system to communicate with the ATmega8U2 on the board. Other Arduino boards use FTDI drivers to communicate with the FTDI chip on the board (or in the USB-serial convertor).

The easiest way to check if the drivers for your board are installed correctly is by opening the **Tools > Serial Port** menu in the Arduino software with the Arduino board connected to your computer. Additional menu items should appear relative to when you open the menu without the Arduino connected to your computer. Note that it shouldn't matter what name the Arduino board's serial port gets assigned as long as that's the one you pick from the menu.

* On Windows 7 (particularly the 64-bit version), you might need to go into the Device Manager and update the drivers for the Uno or Mega 2560. Just right click on the device (the board should be connected to your computer), and point Windows at the appropriate .inf file again. The .inf is in the drivers/ directory of the Arduino software (not in the FTDI USB Drivers sub-directory of it).
* If you get this error when installing the Uno or Mega 2560 drivers on Windows XP: "The system cannot find the file specified", you might try [this suggestion](http://forums.techguy.org/1680041-post1.html) (about adding a "RunOnce" key to "HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion").
* On Linux, the Uno and Mega 2560 show up as devices of the form /dev/ttyACM0. These are not supported by the standard version of the RXTX library that the Arduino software uses for serial communication. The Arduino software download for Linux includes a version of the RXTX library patched to also search for these /dev/ttyACM\* devices. There's also [an Ubuntu package](https://launchpad.net/ubuntu/+source/rxtx/2.2pre2-3) (for 11.04) which includes support for these devices. If, however, you're using the RXTX package from your distribution, you may need to symlink from /dev/ttyACM0 to /dev/ttyUSB0 (for example) so that the serial port appears in the Arduino software.

***Access to the Serial Port***

* On Windows, if the software is slow to start or crashes on launch, or the Tools menu is slow to open, you may need to disable Bluetooth serial ports or other networked COM ports in the Device Manager. The Arduino software scans all the serial (COM) ports on your computer when it starts and when you open the Tools menu, and these networked ports can sometimes cause large delays or crashes.
* Check that you're not running any programs that scan all serial ports, like USB Cellular Wifi Dongle software (e.g. from Sprint or Verizon), PDA sync applications, Bluetooth-USB drivers (e.g. BlueSoleil), virtual daemon tools, etc.
* Make sure you don't have firewall software that blocks access to the serial port (e.g. ZoneAlarm).
* You may need to quit Processing, PD, vvvv, etc. if you're using them to read data over the USB or serial connection to the Arduino board.
* On Linux, you might try running the Arduino software as root, at least temporarily to see if fixes the upload.

***Physical Connection***

* First make sure your board is on (the green LED is on) and connected to the computer.
* The Arduino Uno and Mega 2560 may have trouble connecting to a Mac through a USB hub. If nothing appears in your "Tools > Serial Port" menu, try plugging the board directly to your computer and restarting the Arduino IDE.
* Disconnect digital pins 0 and 1 while uploading as they are shared with serial communication with the computer (they can connected and used after the code has been uploaded).
* Try uploading with nothing connected to the board (apart from the USB cable, of course).
* Make sure the board isn't touching anything metallic or conductive.
* Try a different USB cable; sometimes they don't work.

***Auto-Reset***

* If you have a board that doesn't support auto-reset, be sure that you are resetting the board a couple of seconds before uploading. (The Arduino Diecimila, Duemilanove, and Nano support auto-reset as do the LilyPad, Pro, and Pro Mini with 6-pin programming headers).
* However, note that some Diecimila were accidently burned with the wrong bootloader and may require you to physically press the reset button before uploading; see [this question](http://arduino.cc/en/Guide/Troubleshooting#wrongbootloader) below.
* However, on some computers, you may need to press the reset button on the board after you hit the upload button in the Arduino environment. Try different intervals of time between the two, up to 10 seconds or more.
* If you get this error: [VP 1] Device is not responding correctly. try uploading again (i.e. reset the board and press the download button a second time).

***Bootloader***

* Make sure there's a bootloader burned on your Arduino board. To check, reset the board. The built-in L LED (which is connected to pin 13) should blink. If it doesn't, there may not be a bootloader on your board.

If it still doesn't work, you can ask for help [in the forum](http://www.arduino.cc/cgi-bin/yabb2/YaBB.pl?board=troubleshoot). Please include the following information:

* Your operating system.
* What kind of board you have. If it's a Mini, LilyPad or other board that requires extra wiring, include a photo of your circuit, if possible.
* Whether or not you were ever able to upload to the board. If so, what were you doing with the board before / when it stopped working, and what software have you recently added or removed from your computer?
* The messages displayed when you try to upload with verbose output enabled. To do this, hold down the shift key while clicking on the upload button in the toolbar.

2 **Why do I get "Build folder disappeared or could not be written" (Mac OS X)?**

Did you drag the Arduino.app out of the disk image (and into, say, your Applications folder)? If not, you won't be able to upload the examples.

The Arduino IDE uses the [GNU toolchain](http://en.wikipedia.org/wiki/GNU_toolchain) and AVR Libc to compile programs, and uses avrdude to upload programs to the board. As the Arduino platform uses Atmel microcontrollers, Atmel's development environment, AVR Studio or the newer Atmel Studio, may also be used to develop software for the Arduino.

3 **Why doesn't the Arduino software run after I updated the Java on my Mac?**

The latest Java update from Apple attempts to use 64-bit version of native libraries, but the Arduino application comes with a 32 bit version of the RXTX library. If you launch Arduino, you'll get an error like:

Uncaught exception in main method: java.lang.UnsatisfiedLinkError: /Applications/arduino-0016/Arduino 16.app/Contents/Resources/Java/librxtxSerial.jnilib: no suitable image found. Did find: /Applications/arduino-0016/Arduino 16.app/Contents/Resources/Java/librxtxSerial.jnilib: no matching architecture in universal wrapper

To fix this, click on the Arduino application (e.g. **Arduino 16.app**) in the Finder, and select **Get Info** from the **File**menu. In the info panel, click the **Open in 32 Bit Mode** checkbox. You should then be able to launch Arduino normally.

4 **Why do I get a java.lang.StackOverflowError when I try to compile my program?**

The Arduino environment does some preliminary processing on your sketch by manipulating the code using regular expressions. This sometimes gets confused by certain strings of text. If you see an error like:

java.lang.StackOverflowError

at java.util.Vector.addElement(Unknown Source)

at java.util.Stack.push(Unknown Source)

at com.oroinc.text.regex.Perl5Matcher.\_pushState(Perl5Matcher.java)

or:

at com.oroinc.text.regex.Perl5Matcher.\_match(Perl5Matcher.java)

at com.oroinc.text.regex.Perl5Matcher.\_match(Perl5Matcher.java)

at com.oroinc.text.regex.Perl5Matcher.\_match(Perl5Matcher.java)

at com.oroinc.text.regex.Perl5Matcher.\_match(Perl5Matcher.java)

at com.oroinc.text.regex.Perl5Matcher.\_match(Perl5Matcher.java)

at com.oroinc.text.regex.Perl5Matcher.\_match(Perl5Matcher.java)

at com.oroinc.text.regex.Perl5Matcher.\_match(Perl5Matcher.java)

this is what's happening. Look for unusual sequences involving "double-quotes", "single-quotes", \backslashes, comments, etc. For example, missing quotes can cause problems and so can the sequence '\"' (use '"' instead).

5 **Why doesn't my sketch start when I'm powering the board with an external power supply? (Arduino Diecimila or earlier)**

Because the RX pin is unconnected, the bootloader on the board may be seeing garbage data coming in, meaning that it never times out and starts your sketch. Try tying the RX pin to ground with a 10K resistor (or connecting RX directly to the TX pin).

6 **Why does the Arduino software freeze when I try to upload a program? (on Windows)?**

This might be caused by a conflict with the Logitech process 'LVPrcSrv.exe'. Open the Task Manager and see if this program is running, and if so, kill it before attempting the upload. [More information](http://lists-archives.org/mingw-users/03939-as-exe-hanging-intermittently-solved.html)

7 **What if my board doesn't turn on (the green power LED doesn't light up)?**

If you're using a Decimal or older USB board (e.g. NG), make sure that the jumper (little plastic piece near the USB plug) is on the correct pins. If you're powering the board with an external power supply (plugged into the power plug), the jumper should be on the two pins closest to the power plug. If you're powering the board through the USB, the jumper should be on the two pins closest to the USB plug. This picture shows the arrangment for powering the board from the USB port.

[Attach:jumper.jpg **Δ**](http://arduino.cc/en/Guide/Troubleshooting?action=upload&upname=jumper.jpg)

(Thanks to mrbbp for report and picture)

8 **Why does my Diecimila take such a long time (6-8 seconds) to start my sketch?**

Some of the Arduino Diecimila boards were accidently burned with the Arduino NG boot loader. It should work fine, but has a longer delay when the board is reset (because the NG doesn't have an automatic reset, so you have to time the upload manually). You can recognize the NG boot loader because the LED on pin 13 will blink three times when you reset the board (as compared to once with the Diecimila boot loader). If your Diecimila has the NG boot loader on it, you may need to physically press the reset button on the board before uploading your sketch. You can burn the correct boot loader onto your Diecimila, see the [boot loader](http://arduino.cc/en/Hacking/Bootloader) page for details.

9 **What should I do if I get an error when launching arduino.exe on Windows?**

If you get an error when double-clicking the arduino.exe executable on Windows, for example:

Arduino has encountered a problem and needs to close.

You’ll need to launch Arduino using the run.bat file. Please be patient, the Arduino environment may take some time to open.

10 **Why won't Arduino run on old versions of Mac OS X?**

If you get an error like this:

Link (dyld) error:

dyld: /Applications/arduino-0004/Arduino 04.app/Contents/MacOS/Arduino Undefined symbols:

/Applications/arduino-0004/librxtxSerial.jnilib undefined reference to \_printf$LDBL128 expected to be defined in /usr/lib/libSystem.B.dylib

you probably need to upgrade to Max OS X 10.3.9 or later. Older versions have incompatible versions of some system libraries.

Thanks to Gabe462 for the report.

11 **What do I do if I get an Unsatisfied Link Error error (about native library librxtxSerial.jnilib) when launching Arduino?**

If you get an error like this when launching Arduino:

Uncaught exception in main method: java.lang.UnsatisfiedLinkError: Native Library /Users/anu/Desktop/arduino-0002/librxtxSerial.jnilib already loaded in another classloader

you probably have an old version of the communications library lying around. Search for comm.jar or jcl.jar in /System/Library/Frameworks/JavaVM.framework/ or in directories in your CLASSPATH or PATH environment variables. (reported by Anurag Sehgal)

12 **What about the error “Could not find the main class.”?**

If you get this error when launching Arduino:

Java Virtual Machine Launcher: Could not find the main class. Program will exit.

Make sure that you correctly extracted the contents of the Arduino .zip file – in particular that the **lib** directory is directly inside of the Arduino directory and contains the file **pde.jar**.

13 **What can I do about cygwin conflicts on Windows?**

If you already have cygwin installed on your machine, you might get an error like this when you try to compile a sketch in Arduino:

6 [main] ? (3512) C:\Dev\arduino-0006\tools\avr\bin\avr-gcc.exe: \*\*\* fatal error - C:\Dev\arduino-0006\tools\avr\bin\avr-gcc.exe: \*\*\* system shared memory version mismatch detected - 0x75BE0084/0x75BE009C.

This problem is probably due to using incompatible versions of the cygwin DLL.

Search for cygwin1.dll using the Windows Start->Find/Search facility and delete all but the most recent version. The most recent version \*should\* reside in x:\cygwin\bin, where 'x' is the drive on which you have installed the cygwin distribution. Rebooting is also suggested if you are unable to find another cygwin DLL.

If so, first make sure that you don't have cygwin running when you use Arduino. If that doesn't help, you can try deleting cygwin1.dll from the Arduino directory and replacing it with the cygwin1.dll from your existing cygwin install (probably in c:\cygwin\bin).

*Thanks to karlcswanson for the suggestion.*

14 **Why does do the Arduino software and the Tools menu take a long time to open (on Windows)?**

If the Arduino software takes a long time to start up and appears to freeze when you try to open the Tools menu, there by a conflict with another device on your system. The Arduino software, on startup and when you open the Tools menu, tries to get a list of all the COM ports on your computer. It's possible that a COM port created by one of the devices on your computer slows down this process. Take a look in the Device Manager. Try disabling the devices that provide COM ports (e.g. Bluetooth devices).

15 **Why doesn't my board show in the Tools | Serial Port menu?**

If you're using a USB Arduino board, make sure you installed the FTDI drivers (see the [Howto](http://arduino.cc/en/Guide/Howto) for directions). If you're using a USB-to-Serial adapter with a serial board, make sure you installed its drivers.

Make sure that the board is plugged in: the serial port menu refreshes whenever you open the **Tools** menu, so if you just unplugged the board, it won't be in the menu.

Check that you're not running any programs that scan all serial ports, like PDA sync applications, Bluetooth-USB drivers (e.g. BlueSoleil), virtual daemon tools, etc.

On Windows, the COM port assigned to the board may be too high. From zeveland:

"One little note if you aren't able to export and your USB board is trying to use a high COM port number: try changing the FTDI chip's COM port assignment to a lower one.

"I had a bunch of virtual COM ports set up for Bluetooth so the board was set to use COM17. The IDE wasn't able to find the board so I deleted the other virtual ports in Control Panel (on XP) and moved the FTDI's assignment down to COM2. Make sure to set Arduino to use the new port and good luck."

On the Mac, if you have an old version of the FTDI drivers, you may need to remove them and reinstall the latest version. See [this forum thread for directions](http://www.arduino.cc/cgi-bin/yabb2/YaBB.pl?num=1143197735) (thanks to gck).

16 **What if I get a gnu.io.PortInUseException when uploading code or using the serial monitor (on the Mac)?**

Error inside Serial.<init>()

gnu.io.PortInUseException: Unknown Application

at gnu.io.CommPortIdentifier.open(CommPortIdentifier.java:354)

at processing.app.Serial.<init>(Serial.java:127)

at processing.app.Serial.<init>(Serial.java:72)

This probably means that the port is actually in use by another application. Please make sure that you're not running other programs that access serial or USB ports, like PDA sync application, bluetooth device managers, certain firewalls, etc. Also, note that some programs (e.g. Max/MSP) keep the serial port open even when not using it - you may to need to close any patches that use the serial port or quit the application entirely.

If you get this error with Arduino 0004 or earlier, or with Processing, you'll need to run the macosx\_setup.command, and then restart your computer. Arduino 0004 includes a modified version of this script that all users need to run (even those who ran the one that came with Arduino 0003). You may also need to delete the contents of the **/var/spool/uucp**directory.

17 **I'm having trouble with the FTDI USB drivers.**

Try installing the [latest drivers](http://www.ftdichip.com/Drivers/VCP.htm) from FTDI or contacting their support at support1@ftdichip.com.

18 **Why doesn’t my sketch start when I power up or reset the Arduino board?**

Most likely because you are sending serial data to the board when it firsts turns on. During the first few seconds, the bootloader (a program pre-burned onto the chip on the board) listens for the computer to send it a new sketch to be uploaded to the board. After a few seconds without communication, the bootloader will time out and start the sketch that’s already on the board. If you continue to send data to the bootloader, it will never time out and your sketch will never start. You’ll either need to find a way to stop serial data from arriving for the first few seconds when the board powers (e.g. by enabling the chip that sends the data from within your setup() function) or burn your sketch onto the board with an external, replacing the boot loader.

19 **Why does my sketch appear to upload successfully but not do anything?**

You have selected the wrong item from the Tools > Microcontroller menu. Make sure the selected microcontroller corresponds to the one on your board (either Atmega8 or Atmega168) – the name will be written on the largest chip on the board.

Check for a noisy power supply. It’s possible this could cause the chip to lose its sketch.

Alternatively, the sketch may be too big for the board. When uploading your sketch, Arduino 0004 checks if it’s too big for the Atmega8, but it bases its calculation on a 1 Kb bootloader. You may have a older bootloader that takes up 2 Kb of the 8 Kb of program space (flash) on the Atmega8 instead of the 1 Kb used by the current bootloader. If yours is bigger, only part of the sketch will be uploaded, but the software won’t know, and your board will continually reset, pause, reset.

If you have access to an AVR-ISP or parallel port programmer, you can burn the latest version of the bootloader to your board with the **Tools | Burn Bootloader** menu item. Otherwise, you can tell the Arduino environment the amount of space available for sketches by editing the upload.maximum\_size variable in your preferences file (see: [instructions on finding the file](http://arduino.cc/en/Hacking/Preferences)). Change 7168 to 6144, and the environment should correctly warn you when your sketch is too big.

20 **How can I reduce the size of my sketch?**

The ATmega168 chip on the Arduino board is cheap, but it has only 16 Kb of program code, which isn't very much (and 2 Kb is used by the bootloader).

If you're using floating point, try to rewrite your code with integer math, which should save you about 2 Kb. Delete any **#include** statements at the top of your sketch for libraries that you're not using.

Otherwise, see if you can make your program shorter.

We're always working to reduce the size of the Arduino core to leave more room for your sketches.

21 **Why don’t I get a PWM (an analog output) when I call analogWrite() on pins other than 3, 5, 6, 9, 10, or 11?**

The microcontroller on the Arduino board (the ATmega168) only supports PWM/analogWrite() on certain pins. Calling analogWrite() on any other pins will give high (5 volts) for values greater than 128 and low (0 volts) for values less than 128. (Older Arduino boards with an Atmega8 only support PWM output on pins 9, 10, and 11.)

22 **Why do I get errors about undeclared functions or undeclared types?**

The Arduino environment attempts to automatically generate prototypes for your functions, so that you can order them as you like in your sketch. This process, however, isn't perfect, and sometimes leads to obscure error messages.

If you declare a custom type in your code and create a function that accepts or returns a value of that type, you'll get an error when you try to compile the sketch. This is because the automatically-generated prototype for that function will appear above the type definition.

If you declare a function with a two-word return type (e.g. "unsigned int") the environment will not realize it's a function and will not create a prototype for it. That means you need to provide your own, or place the definition of the function above any calls to it.

23 **Why do I get errors about an invalid device signature when trying to upload a sketch?**

If you get an error like:

avrdude: Yikes! Invalid device signature.

Double check connections and try again, or use -F to override this check.

It can mean one of two things. Either you have the wrong board selected from the Tools > Board menu or you're not using the right version of avrdude. Arduino uses a slightly modified version of avrdude to upload sketches to the Arduino board. The standard version queries for the board's device signature in a way not understood by the bootloader, resulting in this error. Make sure you're using the version of avrdude that comes with Arduino ([source code](http://svn.berlios.de/wsvn/arduino/avrdude-5.4-arduino/#_avrdude-5.4-arduino_)).

Drivers

Drivers provide a way for software on your computer (i.e. the Arduino software) to talk to hardware you connect to your computer (the Arduino board). In the case of Arduino, the drivers work by providing a virtual serial port (or virtual COM port). The Arduino Uno and Mega 2560 use standard drivers (USB CDC) provided by the operating system to communicate with the ATmega8U2 on the board. Other Arduino boards use FTDI drivers to communicate with the FTDI chip on the board (or in the USB-serial convertor).

Arduino Software

* Make sure you have the right item selected in the **Tools > Board** menu. If you have an Arduino Uno, you'll need to choose it. Also, newer Arduino Duemilanove boards come with an ATmega328, while older ones have an ATmega168. To check, read the text on the microcontroller (the larger chip) on your Arduino board. For more information on the board menu items, see the [guide to the Arduino environment](http://arduino.cc/en/Guide/Environment).
* Then, check that the proper port is selected in the **Tools > Serial Port** menu (if your port doesn't appear, try restarting the IDE with the board connected to the computer). On the Mac, the serial port should be something like /dev/tty.usbmodem621 (for the Uno or Mega 2560) or /dev/tty.usbserial-A02f8e (for older, FTDI-based boards). On Linux, it should be /dev/ttyACM0 or similar (for the Uno or Mega 2560) or /dev/ttyUSB0 or similar (for older boards). On Windows, it will be a COM port but you'll need to check in the Device Manager (under Ports) to see which one. If you don't seem to have a serial port for your Arduino board, see the following information about drivers.

*Access to the Serial Port*

* *On Windows, if the software is slow to start or crashes on launch, or the Tools menu is slow to open, you may need to disable Bluetooth serial ports or other networked COM ports in the Device Manager.*

The source code for the following project is:

int Gpsdata; // for incoming serial data

unsigned int finish =0; // indicate end of message

unsigned int pos\_cnt=0; // position counter

unsigned int lat\_cnt=0; // latitude data counter

unsigned int log\_cnt=0; // longitude data counter

unsigned int flg =0; // GPS flag

unsigned int com\_cnt=0; // comma counter

char lat[20]; // latitude array

char lg[20]; // longitude array

void Receive\_GPS\_Data();

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Function : setup()

Description : Use it to initialize variables, pin modes, start using libraries, etc.

The setup function will only run once, after each power up or reset

of the Arduino board.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void setup()

{

Serial.begin(9600); // opens serial port, sets data rate to 9600 bps

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Function : loop()

Description : loops consecutively, allowing your program to change and respond.

Use it to actively control the Arduino board.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void loop()

{

Receive\_GPS\_Data();

Serial.print("Latitude : ");

Serial.println(lat);

Serial.print("Longitude : ");

Serial.println(lg);

finish = 0;pos\_cnt = 0;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Function : Receive\_GPS\_Data()

Description : finding Latitudse and longitude from GPRMC message

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void Receive\_GPS\_Data()

{

while(finish==0){

while(Serial.available()>0){ // Check GPS data

Gpsdata = Serial.read();

flg = 1;

if( Gpsdata=='$' && pos\_cnt == 0) // finding GPRMC header

pos\_cnt=1;

if( Gpsdata=='G' && pos\_cnt == 1)

pos\_cnt=2;

if( Gpsdata=='P' && pos\_cnt == 2)

pos\_cnt=3;

if( Gpsdata=='R' && pos\_cnt == 3)

pos\_cnt=4;

if( Gpsdata=='M' && pos\_cnt == 4)

pos\_cnt=5;

if( Gpsdata=='C' && pos\_cnt==5 )

pos\_cnt=6;

if(pos\_cnt==6 && Gpsdata ==','){ // count commas in message

com\_cnt++;

flg=0;

}

if(com\_cnt==3 && flg==1){

lat[lat\_cnt++] = Gpsdata; // latitude

flg=0;

}

if(com\_cnt==5 && flg==1){

lg[log\_cnt++] = Gpsdata; // Longitude

flg=0;

}

if( Gpsdata == '\*' && com\_cnt >= 5){

com\_cnt = 0; // end of GPRMC message

lat\_cnt = 0;

log\_cnt = 0;

flg = 0;

finish = 1;

}

}

}

}

**CHAPTER 5**

**4.GLOBAL POSITIONING SYSTEM**

**5.1 GPS:**

**GPS** is a space-based [satellite navigation](http://en.wikipedia.org/wiki/Satellite_navigation) system that provides location and time information is all weather conditions, anywhere on or near the Earth where there is an unobstructed line of sight to four or more GPS satellites. The system provides critical capabilities to military, civil and commercial users around the world. It is maintained by the United States government and is freely accessible to anyone with a [GPS receiver](http://en.wikipedia.org/wiki/GPS_receiver).

A GPS receiver calculates its position by precisely timing the signals sent by GPS [satellites](http://en.wikipedia.org/wiki/Satellites) high above the Earth. Each satellite continually transmits messages that include:

* the time the message was transmitted and,
* Satellite position at time of message transmission.

The receiver uses the messages it receives to determine the transit time of each message and computes the distance to each satellite using the speed of light. Each of these distances and satellites' locations defines a sphere. The receiver is on the surface of each of these spheres when the distances and the satellites' locations are correct. These distances and satellites locations are used to compute the location of the receiver using the [navigation equations](http://en.wikipedia.org/wiki/Global_Positioning_System#Navigation_equations). This location is then displayed, perhaps with a [moving map display](http://en.wikipedia.org/wiki/Moving_map_display) or [latitude](http://en.wikipedia.org/wiki/Latitude) and [longitude](http://en.wikipedia.org/wiki/Longitude); elevation or altitude information may be included, based on height above the [geoid](http://en.wikipedia.org/wiki/Geoid) (e.g. [EGM96](http://en.wikipedia.org/wiki/EGM96)).

Basic GPS measurements yield only a position, and neither speed nor direction. However, most GPS units can automatically derive velocity and direction of movement from two or more position measurements. The disadvantage of this principle is that changes in speed or direction can only be computed with a delay, and that derived direction becomes inaccurate when the distance travelled between two position measurements drops below or near the [random error](http://en.wikipedia.org/wiki/Random_error) of position measurement. GPS units can use measurements of the [Doppler shift](http://en.wikipedia.org/wiki/Doppler_shift) of the signals received to compute velocity accurately. More advanced navigation systems use additional sensors like a [compass](http://en.wikipedia.org/wiki/Compass) or an [inertial navigation system](http://en.wikipedia.org/wiki/Inertial_navigation_system) to complement GPS.

The current GPS consists of three major segments. These are the space segment (SS), a control segment (CS), and a user segment (US). The U.S. Air Force develops, maintains, and operates the space and control segments. GPS satellites [broadcast signals](http://en.wikipedia.org/wiki/Broadcast_signal) from space, and each GPS receiver uses these signals to calculate its three-dimensional location (latitude, longitude, and altitude) and the current time.

**5.2 Segment Analysis**

### Space segment:

The space segment (SS) is composed of the orbiting GPS satellites or Space Vehicles (SV) in GPS parlance. The GPS design originally called for 24 SVs, eight each in three approximately circular [orbits](http://en.wikipedia.org/wiki/Orbital_plane_(astronomy)),but this was modified to six orbital planes with four satellites each. The six orbit planes have approximately 55° [inclination](http://en.wikipedia.org/wiki/Inclination) (tilt relative to Earth's [equator](http://en.wikipedia.org/wiki/Equator)) and are separated by 60° [right ascension](http://en.wikipedia.org/wiki/Right_ascension) of the [ascending node](http://en.wikipedia.org/wiki/Orbital_node) (angle along the equator from a reference point to the orbit's intersection). The orbital period is one-half a [sidereal day](http://en.wikipedia.org/wiki/Sidereal_day), i.e., 11 hours and 58 minutes so that the satellites pass over the same locations or almost the same locations every day. The orbits are arranged so that at least six satellites are always within [line of sight](http://en.wikipedia.org/wiki/Line-of-sight_propagation) from almost everywhere on Earth's surface. The result of this objective is that the four satellites are not evenly spaced (90 degrees) apart within each orbit. In general terms, the angular difference between satellites in each orbit is 30, 105, 120, and 105 degrees apart which sum to 360 degrees.

Orbiting at an altitude of approximately 20,200 km (12,600 mi); orbital radius of approximately 26,600 km (16,500 mi), each SV makes two complete orbits each [sidereal day](http://en.wikipedia.org/wiki/Sidereal_day), repeating the same ground track each day. This was very helpful during development because even with only four satellites, correct alignment means all four are visible from one spot for a few hours each day. For military operations, the ground track repeat can be used to ensure good coverage in combat zones.

As of December 2012, there are 32 satellites in the GPS [constellation](http://en.wikipedia.org/wiki/Satellite_constellation). The additional satellites improve the precision of GPS receiver calculations by providing redundant measurements. With the increased number of satellites, the constellation was changed to a no uniform arrangement. Such an arrangement was shown to improve reliability and availability of the system, relative to a uniform system, when multiple satellites fail. About nine satellites are visible from any point on the ground at any one time (see animation at right), ensuring considerable redundancy over the minimum four satellites needed for a position.

### Control segment:

The control segment is composed of:

1. a master control station (MCS),
2. an alternate master control station,
3. four dedicated ground antennas, and
4. six dedicated monitor stations.

The Operation Control Segment (OCS) currently serves as the control segment of record. It provides the operational capability that supports global GPS users and keeps the GPS system operational and performing within specification.

The MCS can also access U.S. Air Force Satellite Control Network (AFSCN) ground antennas (for additional command and control capability) and NGA ([National Geospatial-Intelligence Agency](http://en.wikipedia.org/wiki/National_Geospatial-Intelligence_Agency)) monitor stations. The flight paths of the satellites are tracked by dedicated U.S. Air Force monitoring stations in [Hawaii](http://en.wikipedia.org/wiki/Hawaii), [Kwajalein Atoll](http://en.wikipedia.org/wiki/Kwajalein_Atoll), [Ascension Island](http://en.wikipedia.org/wiki/Ascension_Island), [Diego Garcia](http://en.wikipedia.org/wiki/Diego_Garcia), [Colorado Springs, Colorado](http://en.wikipedia.org/wiki/Colorado_Springs,_Colorado) and [Cape Canaveral](http://en.wikipedia.org/wiki/Cape_Canaveral), along with shared NGA monitor stations operated in England, Argentina, Ecuador, Bahrain, Australia and Washington DC.  The tracking information is sent to the Air Force Space Command MCS at [Schriever Air Force Base](http://en.wikipedia.org/wiki/Schriever_Air_Force_Base) 25 km (16 mi) ESE of Colorado Springs, which is operated by the [2nd Space Operations Squadron](http://en.wikipedia.org/wiki/2nd_Space_Operations_Squadron) (2 SOPS) of the U.S. Air Force. Then 2 SOPS contacts each GPS satellite regularly with a navigational update using dedicated or shared (AFSCN) ground antennas (GPS dedicated ground antennas are located at [Kwajalein](http://en.wikipedia.org/wiki/Kwajalein), [Ascension Island](http://en.wikipedia.org/wiki/Ascension_Island), [Diego Garcia](http://en.wikipedia.org/wiki/Diego_Garcia), and [Cape Canaveral](http://en.wikipedia.org/wiki/Cape_Canaveral)). These updates synchronize the atomic clocks on board the satellites to within a few [nanoseconds](http://en.wikipedia.org/wiki/Nanosecond) of each other, and adjust the [ephemeris](http://en.wikipedia.org/wiki/Ephemeris) of each satellite's internal orbital model. The updates are created by a [Kalman filter](http://en.wikipedia.org/wiki/Kalman_filter) that uses inputs from the ground monitoring stations, [space weather](http://en.wikipedia.org/wiki/Space_weather) information, and various other inputs.

Satellite maneuvers are not precise by GPS standards. So to change the orbit of a satellite, the satellite must be marked *unhealthy*, so receivers will not use it in their calculation. Then the maneuver can be carried out, and the resulting orbit tracked from the ground. Then the new ephemeris is uploaded and the satellite marked healthy again.

OCS successfully replaced the legacy 1970s-era mainframe computer at Schriever Air Force Base in September 2007. After installation, the system helped enable upgrades and provide a foundation for a new security architecture that supported the U.S. armed forces. OCS will continue to be the ground control system of record until the new segment, Next Generation GPS Operation Control System (OCX), is fully developed and functional.

The new capabilities provided by OCX will be the cornerstone for revolutionizing GPS's mission capabilities, and enabling Air Force Space Command to greatly enhance GPS operational services to U.S. combat forces, civil partners and myriad domestic and international users.

The GPS OCX program also will reduce cost, schedule and technical risk. It is designed to provide 50% sustainment cost savings through efficient software architecture and Performance-Based Logistics. In addition, GPS OCX expected to cost millions less than the cost to upgrade OCS while providing four times the capability.

The GPS OCX program represents a critical part of GPS modernization and provides significant information assurance improvements over the current GPS OCS program.

* OCX will have the ability to control and manage GPS legacy satellites as well as the next generation of GPS III satellites, while enabling the full array of military signals.
* Built on a flexible architecture that can rapidly adapt to the changing needs of today's and future GPS users allowing immediate access to GPS data and constellations status through secure, accurate and reliable information.
* Empowers the warfighter with more secure, actionable and predictive information to enhance situational awareness.
* Enables new modernized signals (L1C, L2C, and L5) and has M-code capability, which the legacy system is unable to do.
* Provides significant information assurance improvements over the current program including detecting and preventing cyber-attacks, while isolating, containing and operating during such attacks.
* Supports higher volume near real-time command and control capabilities and abilities.

**User Segment:**

The user segment is composed of hundreds of thousands of U.S. and allied military users of the secure GPS Precise Positioning Service, and tens of millions of civil, commercial and scientific users of the Standard Positioning Service. In general, GPS receivers are composed of an antenna, tuned to the frequencies transmitted by the satellites, receiver-processors, and a highly stable clock (often a [crystal oscillator](http://en.wikipedia.org/wiki/Crystal_oscillator)). They may also include a display for providing location and speed information to the user. A receiver is often described by its number of channels: this signifies how many satellites it can monitor simultaneously. Originally limited to four or five, this has progressively increased over the years so that, as of 2007, receivers typically have between 12 and 20 channels.

GPS receivers may include an input for differential corrections, using the [RTCM](http://en.wikipedia.org/wiki/RTCM) SC-104 format. This is typically in the form of an [RS-232](http://en.wikipedia.org/wiki/RS-232) port at 4,800 bit/s speed. Data is actually sent at a much lower rate, which limits the accuracy of the signal sent using RTCM. Receivers with internal DGPS receivers can outperform those using external RTCM data. As of 2006, even low-cost units commonly include [Wide Area Augmentation System](http://en.wikipedia.org/wiki/Wide_Area_Augmentation_System)(WAAS) receivers.

Many GPS receivers can relay position data to a PC or other device using the [NMEA 0183](http://en.wikipedia.org/wiki/NMEA_0183) protocol. Although this protocol is officially defined by the National Marine Electronics Association (NMEA), references to this protocol have been compiled from public records, allowing open source tools like [gpsd](http://en.wikipedia.org/wiki/Gpsd) to read the protocol without violating [intellectual property](http://en.wikipedia.org/wiki/Intellectual_property) laws. Other proprietary protocols exist as well, such as the [SiRF](http://en.wikipedia.org/wiki/SiRF) and [MTK](http://en.wikipedia.org/wiki/MediaTek) protocols. Receivers can interface with other devices using methods including a serial connection, [USB](http://en.wikipedia.org/wiki/Universal_Serial_Bus), or [Bluetooth](http://en.wikipedia.org/wiki/Bluetooth).

### 5.3 Dedicated GPS navigation devices

Dedicated devices have various degrees of mobility. *Hand-held*, *outdoor*, or *sport* receivers have replaceable batteries that can run them for several hours, making them suitable for [hiking](http://en.wikipedia.org/wiki/Hiking), [bicycle touring](http://en.wikipedia.org/wiki/Bicycle_touring) and other activities far from an electric power source. Their screens are small, and some do not show color, in part to save power. Cases are rugged and some are water resistant.

Other receivers, often called *mobile* are intended primarily for use in a car, but have a small rechargeable internal battery that can power them for an hour or two away from the car. Special purpose devices for use in a car may be permanently installed and depend entirely on the automotive electrical system.

The pre-installed embedded software of early receivers did not display maps; 21st century ones commonly show interactive street maps (of certain regions) that may also show [points of interest](http://en.wikipedia.org/wiki/Point_of_interest), route information and step-by-step routing directions, often in spoken form with a feature called "[text to speech](http://en.wikipedia.org/wiki/Text_to_speech)".

### Mobile phones with GPS capability:

Due in part to regulations encouraging [mobile phone tracking](http://en.wikipedia.org/wiki/Mobile_phone_tracking), including [E911](http://en.wikipedia.org/wiki/E911), the majority of GPS receivers are built into [mobile telephones](http://en.wikipedia.org/wiki/Mobile_telephone), with varying degrees of coverage and user accessibility. Commercial navigation software is available for most 21st-century [smartphones](http://en.wikipedia.org/wiki/Smartphone) as well as some [Java](http://en.wikipedia.org/wiki/Java_(programming_language))-enabled phones that allows them to use an internal or external GPS receiver (in the latter case, connecting via [serial](http://en.wikipedia.org/wiki/Serial_communications) or [Bluetooth](http://en.wikipedia.org/wiki/Bluetooth)). Some phones using [assisted GPS](http://en.wikipedia.org/wiki/Assisted_GPS) (A-GPS) function poorly when out of range of their carrier's cell towers. Others can navigate worldwide with satellite GPS signals as well as a dedicated portable GPS receiver does, upgrading their operation to A-GPS mode when in range. Still others have a [hybrid positioning system](http://en.wikipedia.org/wiki/Hybrid_positioning_system) that can use other signals when GPS signals are inadequate.

More [bespoke](http://en.wikipedia.org/wiki/Bespoke) solutions also exist for [smartphones](http://en.wikipedia.org/wiki/Smartphone) with inbuilt GPS capabilities. Some such phones can use [tethering](http://en.wikipedia.org/wiki/Tethering) to double as a [wireless modem](http://en.wikipedia.org/wiki/Wireless_modem) for a [laptop](http://en.wikipedia.org/wiki/Laptop), while allowing GPS-navigation/localization as well. One such example is marketed by [Verizon Wireless](http://en.wikipedia.org/wiki/Verizon_Wireless) in the [United States](http://en.wikipedia.org/wiki/United_States), and is called [VZ Navigator](http://en.wikipedia.org/wiki/VZ_Navigator). The system uses [gps One](http://en.wikipedia.org/wiki/GpsOne) technology to determine the location, and then uses the mobile phone's data connection to download maps and calculate navigational routes. Other products including [iPhone](http://en.wikipedia.org/wiki/IPhone) are used to provide similar services. [Nokia](http://en.wikipedia.org/wiki/Nokia) gives [Ovi Maps](http://en.wikipedia.org/wiki/Ovi_Maps) free on its smartphones and maps can be preloaded.

According to market research from the independent analyst firm Berg Insight, the sales of GPS-enabled GSM/WCDMA handsets was 150 million units in 2009, while only 40 million separate GPS receivers were sold.

GPS navigation applications for mobile phones include on-line (e.g. [Waze](http://en.wikipedia.org/wiki/Waze), [Google Maps Navigation](http://en.wikipedia.org/wiki/Google_Maps_Navigation)) and off-line (e.g. [iGo](http://en.wikipedia.org/wiki/IGO_(software)) for Android, Maverick) navigation applications. [Google Maps Navigation](http://en.wikipedia.org/wiki/Google_Maps_Navigation), which is included with [Android](http://en.wikipedia.org/wiki/Android_(operating_system)), means most smartphone users only need their phone to have a [personal navigation assistant](http://en.wikipedia.org/wiki/Personal_navigation_assistant).

Many Android smartphones have an additional GPS feature, called **EPO** ([Extended Prediction Orbit](http://en.wikipedia.org/w/index.php?title=Extended_Prediction_Orbit&action=edit&redlink=1)). The phone downloads a file to help it locate GPS satellites more quickly and reduce the Time to First Fix.

### Laptop PC GPS:

Software companies have made available [GPS navigation software](http://en.wikipedia.org/wiki/GPS_navigation_software) programs for in-vehicle use on laptop computers. Benefits of GPS on a laptop include larger map overview, ability to use the keyboard to control GPS functions, and some GPS software for laptops offers advanced trip-planning features not available on other platforms, such as midway stops, capability of finding alternative scenic routes as well as only highway option.

Consumer GPS navigation devices include:

* Dedicated GPS navigation devices
* GPS modules that need to be connected to a computer to be used
* GPS loggers that record trip information for download. Such [GPS tracking](http://en.wikipedia.org/wiki/GPS_tracking) is useful for trailblazing, mapping by hikers and cyclists, and the production of [geocoded photographs](http://en.wikipedia.org/wiki/Geocoded_photograph).
* Converged devices, including GPS Phones and [GPS cameras](http://en.wikipedia.org/wiki/Auto-geotagging), in which GPS is a feature rather than the main purpose of the device. Those devices are the majority, and may use [assisted GPS](http://en.wikipedia.org/wiki/Assisted_GPS) or standalone (not network dependent) or both. The vulnerability of consumer GPS to radio frequency interference from [planned wireless data services](http://en.wikipedia.org/wiki/LightSquared#Interference_issues) is controversial.

### GPS modules:

Other GPS devices need to be connected to a computer in order to work. This computer can be a [home computer](http://en.wikipedia.org/wiki/Home_computer), [laptop](http://en.wikipedia.org/wiki/Laptop), [PDA](http://en.wikipedia.org/wiki/Personal_digital_assistant), [digital camera](http://en.wikipedia.org/wiki/Digital_camera), or smartphones. Depending on the type of computer and available connectors, connections can be made through a [serial](http://en.wikipedia.org/wiki/Serial_cable) or [USB](http://en.wikipedia.org/wiki/Universal_Serial_Bus) cable, as well as Bluetooth, [CompactFlash](http://en.wikipedia.org/wiki/CompactFlash), [SD](http://en.wikipedia.org/wiki/Secure_Digital_card), [PCMCIA](http://en.wikipedia.org/wiki/PC_Card) and the newer [Express Card](http://en.wikipedia.org/wiki/ExpressCard). Some PCMCIA/Express Card GPS units also include a [wireless modem](http://en.wikipedia.org/wiki/Wireless_modem).

Devices usually do not come with pre-installed [GPS navigation software](http://en.wikipedia.org/wiki/GPS_navigation_software), thus, once purchased, the user must install or write their own software. As the user can choose which software to use, it can be better matched to their personal taste. It is very common for a PC-based GPS receiver to come bundled with a navigation software suite. Also, GPS modules are significantly cheaper than complete stand-alone systems (around [€](http://en.wikipedia.org/wiki/Euro)50 to €100). The software may include maps only for a particular region, or the entire world, if software such as Google Maps, [Networks in Motion's Atlas Book mobile navigation platform](http://en.wikipedia.org/wiki/VZ_Navigator), etc., are used.

Some hobbyists have also made some GPS devices and open-sourced the plans. Examples include the [Elektor GPS units](http://www.elektor.com/magazines/2008/october/multi-purpose-gps-receiver.684158.lynkx). These are based around a [SiRFstarIII](http://en.wikipedia.org/wiki/SiRFstarIII) chip and are comparable to their commercial counterparts. Other chips are also available.

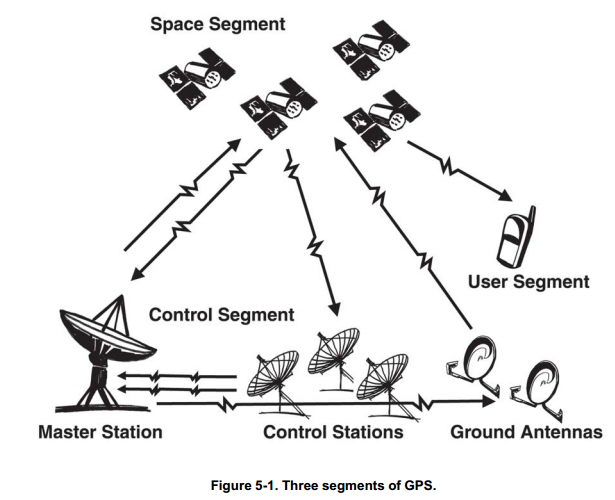
### Surveillance:

Privacy concerns also arise when employers use [GPS tracking units](http://en.wikipedia.org/wiki/GPS_tracking_unit) to track their employees' location, for example using [vehicle tracking systems](http://en.wikipedia.org/wiki/Vehicle_tracking_system). This raises a major question about whether this violates personal privacy of employees. It raises a lot more concern for privacy violation if the employers collect geo-location data of their employee after work hours and during their holidays. In 2010, [New York Civil Liberties Union](http://en.wikipedia.org/wiki/New_York_Civil_Liberties_Union) filed a case against the Labor Department for firing Michael Cunningham after tracking Michael Cunningham's daily activity and locations using GPS device that has attached in his car. This raises few questions regarding the limit of surveillance. The worst privacy violation is done by FBI when they tracked down Antoine Jones GPS devices even without any search warrants. Later the Federal Appeal Court rejected FBI's surveillance data as a proof against Antoine Jones.

Surveillance open source GPS with Filters(like night vision, Sharpening location, Contrast) with optimal route finding algorithm embedded in the application developed in java.

**5.4** Working of Global Positioning System

The basis of the GPS is a constellation of satellites that are continuously orbiting the earth. These satellites, which are equipped with atomic clocks, transmit radio signals that contain their exact location, time and other information. The radio signals from the satellites, which are monitored and corrected by control stations, are picked up by the GPS receiver. A GPS receiver needs only three satellites to plot a rough, 2D position, which will not be very accurate. Ideally, four or more satellites are needed to plot a 3D position, which is much more accurate.

****

**Position determination using GPS:**

The GPS receiver uses the following information to determine a position.

* **Precise location of satellites:**

When a GPS receiver is first turned on, it downloads orbit information from all the satellites called an almanac. This process, the first time, can take as long as 12 minutes; but once this information is downloaded; it is stored in the receiver’s memory for future use.

* **Distance from each satellite:**

The GPS receiver calculates the distance from each satellite to the receiver by using the distance formula: distance = velocity x time. The receiver already knows the velocity, which is the speed of a radio wave or 186,000 miles per second (the speed of light). To determine the time part of the formula, the receiver times how long it takes for a signal from the satellite to arrive at the receiver. The GPS receiver multiplies the velocity of the transmitted signal by the time it takes the signal to reach the receiver to determine distance.

* **Triangulation to determine position:**

The receiver determines position by using triangulation. When it receives signals from at least three satellites the receiver should be able to calculate its approximate position (a 2D position). The receiver needs at least four or more satellites to calculate a more accurate 3D position. The position can be reported in latitude/longitude, UTM, or other coordinate system.

**Sources of Errors**

The GPS is not a perfect system. There are several different types of errors that can occur when using a GPS receiver, for example:

* **User mistakes**

User mistakes account for most GPS errors; and a GPS receiver has no way to identify and correct these mistakes. Common examples of user mistakes include:

– Inputting incorrect information into a GPS receiver, such as the datum, and when creating a waypoint.

– Unknowingly relying on a 2D position instead of a 3D position for determining position

coordinates. This mistake can result in distance errors in excess of a mile. The signal from the satellite may be blocked by buildings, terrain, electronic interference, and sometimes dense foliage. A GPS receiver needs a fairly clear view of the sky to operate.

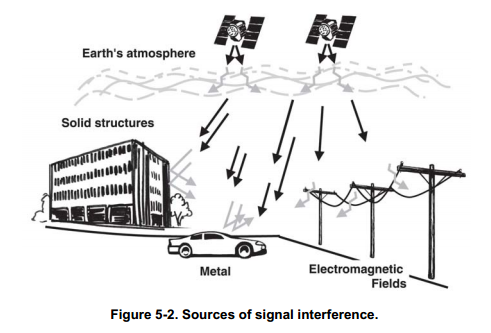
– The human body can cause signal interference. Holding a GPS receiver close to the body can block some satellite signals and hinder accurate positioning. If a GPS receiver must be hand held without benefit of an external antenna, facing to the south can help to alleviate signal blockage caused by the body because the majority of GPS satellites are oriented more in the earth’s southern hemisphere.

* **Multipath interference**

Multipath interference is caused by the satellite signal reflecting off of vehicles, buildings, power lines, water and other interfering objects (Figure 5-2). Multipath is difficult to detect and sometimes impossible for the user to avoid or for the receiver to correct. When using a GPS receiver in a vehicle place the external antenna on the roof o the vehicle to eliminate most signal interference caused by the vehicle. If the GPS receiver is placed on the dashboard there will always be some multipath interference.

* **Satellite and receiver clock errors**

These can be slight discrepancies in the satellite’s atomic clocks which may cause slight position errors in the GPS receiver. Errors are monitored and corrected by the Master Control Station.



* **Orbit errors**

Satellite orbit pertains to the altitude, position, and speed of the satellite. Satellite orbits vary due to gravitational pull and solar pressure fluctuations. Orbit errors are also monitored and corrected by the Master Control Station.

* **Satellite geometry**

The location of GPS satellites in relation to a GPS receiver on the ground can impact the receiver’ ability to triangulate a 3D position. The quality of a receiver’s triangulated position improves the further apart GPS satellites are located from each other in the sky above the receiver. The quality decreases if the satellites are grouped close together in the sky above the receiver.

* **Atmospheric interference**

The atmosphere can slow or speed up the satellite signal. Fortunately, error caused by atmospheric conditions (ionized air, humidity, temperature, pressure) has been reduced with the implementation of the Wide Area Augmentation System (WAAS).

* **Selective Availability**

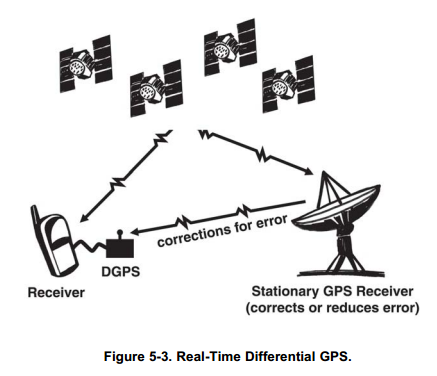
Selective Availability is the intentional degradation (limits accuracy of satellite signals) of the GPS system by the U.S. Department of Defense for security reasons. At this time there is no Selective Availability in force; however, it can be reactivated without notice to GPS users.

* **Correction systems**

Correction systems have been designed to reduce some of the sources of error with GPS.

* **Real-time Differential GPS**

Real-time Differential GPS (DGPS) employs a second, stationary GPS receiver at a precisely measured spot, usually established through traditional survey methods (Figure 5-3). This receiver corrects or reduces errors found in the GPS signals, including atmospheric distortion, orbital anomalies, Selective Availability (when it existed), and other errors. A DGPS station is able to do this because its computer already knows its precise location, and can easily determine the amount of error provided by the GPS signals. DGPS cannot correct for GPS receiver noise in the user’s receiver, multipath interference, and user mistakes. In order for DGPS to work properly, both the user’s receiver and the DGPS station receiver must be accessing the same satellite signals at the same time.



* **Wide Area Augmentation System**

The Wide Area Augmentation System (WAAS) is an experimental system designed to enhance and improve aircraft flight approaches using GPS and WAAS satellites. The WAAS can be considered an advanced real-time differential GPS. It uses its own geo-stationary satellites positioned over the equator to transmit corrected GPS signals to receivers capable of receiving these signals. Problems with WAAS include poor signal reception under dense tree canopy and in canyons, as well as decreased capability in northerly latitudes. Many GPS receivers are now capable of receiving the WAAS signal. However, WAAS should not be considered a consistently reliable source for improving the accuracy of GPS until the technology improves.

**5.5 Using a GPS Receiver**

There are several different models and types of GPS receivers. Refer to the owner’s manual for your GPS receiver and practice using it to become proficient.

When working on an incident with a GPS receiver it is important to:

* Always have a compass and a map.
* Have a GPS download cable.
* Have extra batteries.
* Know memory capacity of the GPS receiver to prevent loss of data, decrease in accuracy of data, or other problems.
* Use an external antennae whenever possible, especially under tree canopy, in canyons, or while flying or driving.
* Set up GPS receiver according to incident or agency standard regulation; coordinate system.
* Take notes that describe what you are saving in the receiver.

**Inputs**

Each time you use a GPS receiver, you will need to input information such as:

* Position format units (example: UTM 11T 0557442m E 4836621m N).
* This input determines the way positions are displayed on the receiver screen. For example, sometimes you may want to use latitude/longitude coordinates and other times it may be better to use UTM coordinates.
* Map datum (example: WGS 84, NAD 27 and NAD 83).

This input ensures that your GPS receiver and map are both using the same datum, which is extremely important for accuracy.

* Distance units (feet, miles, meters).
* Elevation units (feet or meters).
* North reference (true, magnetic, or grid) and Time format (12 or 24 hour) and time zone.

**Waypoints**

A waypoint is a position based on geographic coordinate values, such as latitude/longitude and UTM, stored in the GPS receiver’s memory. They are sometimes referred to as landmarks. Once the waypoint is saved it remains static in the GPS receiver’s memory until edited or deleted.

**How Waypoints are Determined**

A waypoint can either be a saved position fix or can be created by manually entering coordinates into the receiver.

* To turn a position location into a waypoint is simply a matter of saving the receiver’s current position as a waypoint. The receiver will give the position coordinates an alpha-numeric name or the user can designate a name. Once a position fix is saved, it become a waypoint with static coordinates saved in the receiver’s memory.
* Users can enter waypoints into the GPS receiver. For example, coordinates on a map or coordinates radioed in from a remote location can be entered into a GPS receiver.

**Naming Waypoints**

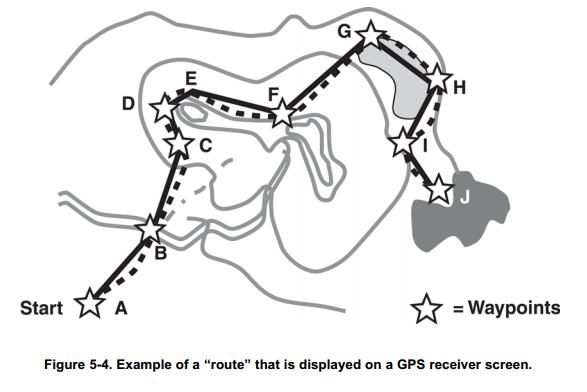
The GPS receiver will automatically name waypoints with an alpha-numeric name; however, it is best if you designate a unique name for each waypoint so you will know exactly what the waypoint is referring to. Use short descriptive designations because long names can be hard to read when they are downloaded. Some possible designations include:

* D1, D2, D3 for different dozer lines
* HL1, HL2, HL3 for different hand lines.
* DP1, DP2, DP3 for different drop points.
* H1, H2, H3 for different heli-spots.
* A1, A2, A3 for different access points.

**Routes**

Routes are just a sequence of waypoints (Figure 5-4). When navigating a route, the GPS receiver will automatically change the destination waypoint to the next waypoint on the list as it reaches each waypoint.

Once one waypoint is passed, the GPS receiver will navigate to the next waypoint. When a route is first activated, the GPS receiver will assume that the first leg is A to B. B is the waypoint being navigated to and A is the anchor point that defines the first leg of the route.



**Terminology**

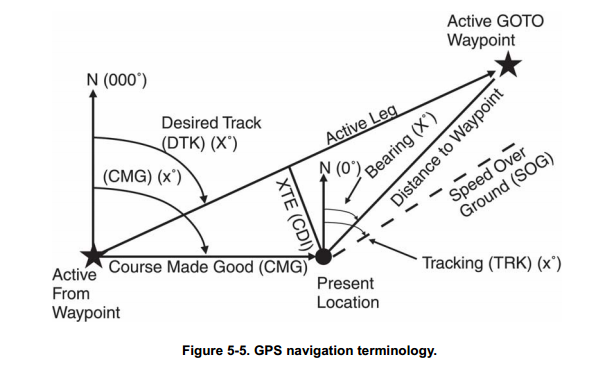
There is a lot of terminology associated with using a GPS receiver.

* **Active from Waypoint**

This is the starting waypoint or the receiver’s last waypoint in an active route.

* **Active GOTO Waypoint**

This is the designated destination in the receiver, whether in an active route or as a single waypoint.

****

* **Active Leg**

Active leg is always a straight line between the last waypoint and the GOTO waypoint. A GPS receiver always plots the most efficient, straight-line course of travel between two points – the active leg. If the receiver is following a route, the active leg will be the desired track between the last waypoint in the route, and the next waypoint in the route. If the receiver has deviated from the route, the receiver selects the closest leg to its position and makes it the active leg in the route (the next waypoint in the route list becomes the GOTO destination waypoint).

* **Bearing (BRG)/Desired Track (DTK)**

In GPS the term bearing is used instead of azimuth. As used in GPS, bearing is the compass direction (expressed in degrees) from the present position to desired destination waypoint or the compass direction between any two waypoints.

* **Course Deviation Indicator (CDI)**

This graphically shows the amount and direction of Crosstrack Error.

* **Course Made Good (CMG) or Course Over Ground (COG)**

This is the present direction of travel expressed in degrees from north. It is not necessarily the most direct path.

* **Crosstrack Error (XTE)**

This is the distance off the desired track (active leg) on either side of the active leg. It’s the linear difference between the Desired Track (DTK) and your actual Course Made Good (CMG).

* **Desired Track (DTK)**

This is a function of GOTO. It is shown in degrees from north. DTK is measured along the active leg (a straight line between two waypoints in a route) or from your current position to a designated GOTO waypoint, when not navigating a route.

* **Estimated Position Error (EPE)**

A measurement of horizontal position error in feet or meters based upon a variety of factors including dilution of precision (DOP) and satellite signal quality.

* **Estimated Time En Route (ETE)**

The time left to destination based upon present speed and course.

* **Estimated Time of Arrival (ETA)**

The time of day of arrival at a destination

* **Fix**

A single position with latitude, longitude (or grid position), altitude, time, and date.

* **GOTO Function**

The GOTO function gives GPS receivers the capability of leading a person to any specified place. Simply enter the coordinate of desired destination into the GPS receiver as a waypoint and then, by using the GOTO function, tell the receiver to guide to destination. The receiver guides to destination using a steering screen. There are several different versions of a steering screen, but they all point in the direction needed to travel to from present position to the waypoint selected.

* **GOTO Waypoint**

If traveling from one waypoint to another (using GOTO), then XTE will show the distance of deviation of your actual route from the active leg (a straight line) between those waypoints.

* **Speed Over Ground (SOG)**

This is the velocity you are traveling.

* **Tracking (TRK)/ Heading (HDG)**

This is the direction you are actually traveling or heading, expressed in degrees from north.

* **Track Log**

A track log is the GPS unit’s record of travel or where you have been. As you move along, your every movement is being stored. Receivers with a TracBack feature will allow you to reverse your route taking you back the same way you originally traveled. As you move along, most GPS receivers show your track on a map screen.

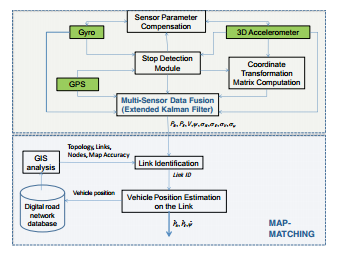
* **Velocity Made Good (VMG)**

Velocity made good is the speed at which the destination is approached. If you are directly on course, VMG is the same value as SOG, but if you stray from course, VMG decreases and is less than SOG.

**5.6 Uninterrupted Portable Car Navigation System using GPS, Map and Inertial Sensors Data**

The approach uses digital maps, 3D accelerometer and one gyro for directional measurements to improve positioning availability and reliability in weak signal environment and during short GPS signal outages. This system does not require vehicle installation and can be easily transferred between vehicles. Loosely coupled extended Kalman ﬁlter and probabilistic map-matching algorithm provide optimally tuned navigation solution and continuous auto calibration of inertial sensors. The system accuracy performance was investigated using ﬁeld tests in an urban environment.

Accurate and continuous position calculation is a key task for vehicle navigation and telematics applications. In most portable car navigation and telematics devices, the position is calculated based only on GPS data.



Accurate and uninterrupted position calculation is a key task for vehicle navigation and telematics applications. In most portable car navigation and telematics devices the position is calculated based only on GPS data. However, in urban canyons stand-alone GPS suffers signal masking and reﬂections of the signal from buildings, large vehicles, and other reﬂective surfaces.

Both the gyro and accelerometers satisfy the requirements for mass market portable consumer devices: low cost, light weight, small power consumption. An odometer is not used because this requires additional car installation; our system is intended mainly for portable devices such as mobile phones, GPS-based peripherals and handheld GPS navigation devices.

**Integration concept**

The proposed car navigation system includes different types of navigation sensors and technologies.

This methodology comprises several stages of data processing:

* Inertial sensors data for stop detection
* Inertial sensor data for position, velocity and heading computations (DR computations)
* Calibration of inertial sensors when vehicle is stationary
* Integrated GPS/DR navigation
* Map-matching

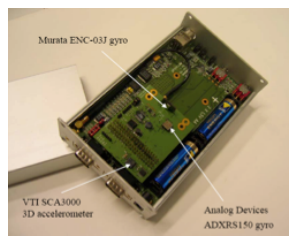
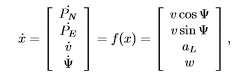


Fig 5.6 Integration concept reg Map-matching and kalman filter

**5.7 Kalman Filter:**

The real time data fusion algorithm employs an extended Kalman ﬁlter to combine computed GPS position, velocity, and heading with the acceleration and heading rate measure- ments provided by the 3D accelerometer and heading gyro. The EKF uses the values of the ﬁlter states to predict the future states through a dynamic model which is based on dead reckoning equations to estimate position, velocity, and heading.



where PN and PE are the vehicle north and east positions, respectively, Ψ is the vehicle heading, w is the measured heading rate, v is the speed over ground, and aL is the measured vehicle acceleration in the longitudinal direction.

The vehicle frame longitudinal acceleration is calculated by transforming three-dimensional acceleration vector measured by accelerometers in the sensor frame into vehicle frame and calculating projection of the transformed vector on vehicles longitudinal axes. It also includes the effect of gravitational forces because of unknown road grade.



where a is the vehicle longitudinal acceleration, g is the gravitational constant, θ is the road grade and bL is the longitudinal acceleration error. To properly model accelerometer and gyro measurement errors and unknown road grade the state vector is augmented with two additional states: gyro bias, δw, and acceleration bias and misalignment, δa, which includes unknown road grade.

The observation vector is calculated by taking the difference between GPS and DR corresponding positions and velocities, and in some cases heading. The measurement update can take the following forms:

• vehicle position, velocity, and heading,

• vehicle position and velocity,

• or velocity, only in the case of the zero velocity update (ZUPT).



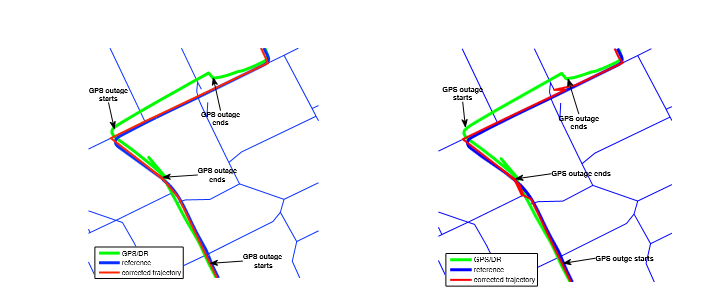
The position accuracy of a single frequency L1 GPS receiver is approximately 10 m in the horizontal axis and 15 m in the vertical axis. A single frequency L1 GPS receiver determines velocity based on the Doppler shift of the GPS carrier wave. The velocity accuracy in the horizontal axis can reach 2-5 cm/s and in the vertical axis 4-10 cm/s 1-σ standard deviation of the stochastic errors . The accuracy of GPS depends heavily on satellite geometry and multipath errors. In this project, the update rate of the GPS receiver was set to 1 Hz to reduce the amount of computations in the processor. The GPS velocity measurements can be also used to determine vehicle heading.



where vGPS E and vGPS N are the east and north GPS velocity measurements, respectively.

However the above expression can be approximated to





Reference trajectory, uncorrected GPS/DR trajectory, and GPS/DR trajectory corrected using our map-matching particle ﬁlter and Reference trajectory, uncorrected GPS/DR trajectory, and GPS/DR trajectory corrected using Mahalanobis map-matching scheme.

**MAP-Matching**

The digital road network map is another important component of our positioning system. A map database is a source of valuable information that can be used to improve the accuracy of the position given by the GPS/DR navigation system and calibrate the DR sensors. Map-matching algorithms usually consist of two steps: identiﬁcation of the road link where vehicle is most likely travelling and estimation of vehicle position on the selected road link.

This approach is mainly based on the proximity between the position ﬁx and the road, the difference between the estimated vehicle heading and the current segment heading from the map data, and vehicles position, velocity and heading prior to the current moment. The determination of the vehicle location on the segment is a challenging task, especially considering the errors associated with both the digital map and navigation sensors. When the vehicle is travelling on a straight road the orthogonal projection of the position ﬁx onto the selected road segment is used to calculate the vehicle location on the segment. In this case only cross track error can be deduced by map-matching.

Our map-matching algorithm is built on the Bayesian frame- work and uses particle ﬁltering. Because it is Bayesian, it does not associate a single speciﬁc street to a position ﬁx, but computes a probability for every street on the map, and it is incremental, that is, it processes one single GPS/DR ﬁx at a time rather than trying to identify a trajectory on the map from a sequence of ﬁxes.

Three major factors that influence the Map matching techniques are :

* Model
* Particle filtering
* Position accuracy enhancement

In the above figure the red line corresponds to a corrected solution after map-matching algorithm was applied.

We have implemented two different versions of map-matching algorithm: one is based on probabilistic approach implemented in a form of particle ﬁlter, another is based on calculation of Mahalanobis distance between road segments and a vehicle.

The performance of particle ﬁlter based algorithm is superior to the second approach. This test shows that the combined GPS/DR solution provides signiﬁcant improvement; accurate position, velocity, and heading information were available even in the absence of GPS signal.

**CONCLUSION and FUTURE SCOPE**

Low-cost inertial sensors and map- matching algorithm can signiﬁcantly improve GPS positioning by continuing to output position during short GPS outages with sufﬁcient accuracy for most of car navigation applications. The integrated GPS+MEMS system has also demonstrated improvement of position and velocity accuracy in high mul- tipath urban canyon environment and the ability to provide continuous output of the vehicle heading even when vehicle is not moving. This is useful when we apply the map-matching algorithm. This device does not require any installation in the vehicle. It works in all vehicles and can be easily transferred between vehicles. Finally, it should be noted that our design is suitable for portable navigation devices since the cost, size and power consumption of inertial sensors meet the requirements for mass market consumer electronics.low-cost inertial sensors and map- matching algorithm can signiﬁcantly improve GPS positioning by continuing to output position during short GPS outages with sufﬁcient accuracy for most of car navigation applications. The integrated GPS+MEMS system has also demonstrated improvement of position and velocity accuracy in high mul- tipath urban canyon environment and the ability to provide continuous output of the vehicle heading even when vehicle is not moving. This is useful when we apply the map-matching algorithm. This device does not require any installation in the vehicle. It works in all vehicles and can be easily transferred between vehicles. Finally, it should be noted that our design is suitable for portable navigation devices since the cost, size and power consumption of inertial sensors meet the requirements for mass market consumer electronics.

When stand-alone GPS ﬁxes (with SA on) are utilized, the calibration error induced by SA when the position ﬁxes are available has very signiﬁcant deleterious effects on system performance after the position ﬁxes become unavailable. For a system comprised of a GPS receiver, a rate gyro, and an odometer, improving gyro quality signiﬁcantly does not cause a commensurate improvement in system performance because heading error is strongly affected by SA. For this system, then, calibration error plays a more signiﬁcant role than sensor quality in determining system performance after position ﬁxes become unavailable.

**BIBLIOGRAPHY**

<http://arduino.cc>

<http://www.wikipedia.com>

http://www.alldatasheets.com

[Example of hook-up of GPS-phone as wireless modem and GPS receiver](http://www.sonyericsson.com/cws/products/pccards/overview/ec400g?cc=be&lc=en)

[**Jump up ^**](http://en.wikipedia.org/wiki/GPS_navigation_device#cite_ref-2) ["GPS and Mobile Handsets - 4th edition"](http://www.berginsight.com/ReportPDF/Summary/bi-gps4-sum.pdf) (PDF). Retrieved 2012-02-01

[**Jump up ^**](http://en.wikipedia.org/wiki/GPS_navigation_device#cite_ref-3) [Kevin J. O'Brien, New York Times, 2010 Nov 15](http://www.nytimes.com/2010/11/15/technology/15iht-navigate.html) Smartphone Sales Taking Toll on G.P.S. Devices

[**Jump up ^**](http://en.wikipedia.org/wiki/GPS_navigation_device#cite_ref-4) [Extended Prediction Orbit](http://www.bt747.org/book/agps) GPS data logger software

[**Jump up ^**](http://en.wikipedia.org/wiki/GPS_navigation_device#cite_ref-5) ["List of laptop GPS navigation software programs and reviews"](http://www.laptopgpsworld.com/37-list-laptop-gps-navigation-software-programs-reviews#post73). Laptopgpsworld.com

X. Zhang, Q. Wang, and D. Wan, “Map matching in road crossings of ur- ban canyons based on road traverses and linear heading-change model,” Instrumentation and Measurement, IEEE Transactions on, vol. 56, no. 6, pp. 2795–2803, Dec. 2007.

M. Chowdhary, J. Colley, and M. Chansarkar, “Improving GPS location availability and reliability by using a suboptimal, low-cost MEMS sensor set,” Proceedings of ION GPS, 2007.

N. El-Sheimy, “The potential of partial IMUs for land vehicle naviga- tion,” Inside GNSS, 2008.

ADXRS150 150/s Single Chip Yaw Rate Gyro with Signal Conditioning, Analog Devices Inc. [Online]. Available: <http://www.analog.com/en/prod/0,2877,ADXRS150,00.html>

Atheros, Atheros Communications. [Online]. Available: <http://www.atheros.com>