Mobile phone jammer Lloydminster , mobile phone jammer circuit diagram pdf

- advanced mobile phone signal jammer with highlow o
- advantages of mobile phone jammer
- buy mobile phone jammer
- electronic mobile phone jammer
- gps mobile phone jammer abstract judgment
- gps mobile phone jammer abstract request
- gps mobile phone jammer factory
- gps mobile phone jammer for sale
- gps mobile phone jammer laws
- how can i make a mobile phone jammer
- mini portable mobile phone signal jammer
- mobile phone jammer Manitoba
- mobile phone jammer New Brunswick
- mobile phone and gps jammer china
- mobile phone gps jammer app
- mobile phone gps jammer yakima
- mobile phone jammer australia
- mobile phone jammer circuit pdf
- mobile phone jammer cost
- mobile phone jammer dealers
- mobile phone jammer dealers in kerala
- mobile phone jammer detector
- mobile phone jammer Dieppe
- mobile phone jammer for home
- mobile phone jammer in hyderabad
- mobile phone jammer in uk
- mobile phone jammer ireland
- mobile phone jammer Kawartha Lakes
- mobile phone jammer manufacturer
- mobile phone jammer Melville
- mobile phone jammer Mercier
- mobile phone jammer Nottingham
- mobile phone jammer overview
- mobile phone jammer Penticton

- mobile phone jammer Port Colborne
- mobile phone jammer price in india
- mobile phone jammer Prince Edward County
- mobile phone jammer Prince Rupert
- mobile phone jammer Steinbach
- mobile phone jammer Thurso
- mobile phone jammer Trail
- mobile phone jammer York
- mobile phone jammers in pakistan
- mobile phone signal jammer with pre scheduled time
- mobile phone signal jammer with remote control
- mobilephonejammers
- office mobile phone jammer
- phone mobile jammer yakima
- raspberry pi mobile phone jammer
- where can i get a mobile phone jammer

Permanent Link to Innovation: Python GNSS Receiver
2021/04/09
An Object-Oriented Software Platform Suitable for Multiple Receivers By Eliot Wycoff, Yuting Ng, and Grace Xingxin Gao INNOVATION INSIGHTS by Richard Langley AND NOW FOR SOMETHING COMPLETELY DIFFERENT. My first introduction to computer programming was during a visit to the Faculty of Mathematics at the University of Waterloo when I was still a high school student. We got to keypunch a simple program onto cards using the FORTRAN programming language and submit the "job" to the university's IBM 7040 mainframe computer. That visit helped seal the choice of Waterloo for my undergraduate education — but in applied physics, not math. Once I became an undergraduate, I learned how to properly program in FORTRAN (actually FORTRAN IV with the WATFOR compiler developed at Waterloo) and in assembly language on the SPECTRE virtual computer (written in FORTRAN), both on Waterloo's new IBM 360 mainframe. Knowing how to program was instrumental in my graduate work on the geodetic application of very long baseline interferometry (VLBI) at York University. Being humble Canadians (and despite the fact that VLBI was invented in Canada), we called it just LBI. My LBI data analysis FORTRAN program was initially on a box full of punched cards that I would have to carry back and forth to the computer center being careful not to drop the box and get the cards out of order.   While I was a graduate student, I also got to use the Spiras-65 minicomputer that controlled the playback of the LBI recorded tapes at the National Research Council in Ottawa. It was programmed using punched paper tape. I saw the progression from punched tape and cards to the use of terminals to enter programs and magnetic tapes for storing them and the data to be analyzed. The University of New Brunswick, where I came to work in 1981, was one of the first universities in Canada to introduce an interactive terminal- (or work-station-) based time-sharing system for programmers to develop and run their jobs on the central computer. The last card reader at UNB was retired in 1987. By the time I came to work at UNB, the era of the personal computer had already dawned. Although the Department of Surveying Engineering (as it was then called) acquired an HP 1000

minicomputer for various research tasks, personal computers began to show up on faculty members' desks and in their labs. Some of us started out with Apple II computers (we used them, for example, for recording data from Transit–U.S. Navy Navigation Satellite System–receivers) and progressed through various Macintosh models. Once I became a professor, I did less and less programming myself–leaving it up to my graduate students to do the heavy lifting in that area. These days, my personal programming efforts are limited to short scripts mostly using the Python language. Python, which gets its name from the Monty Python's Flying Circus television series, was first introduced back in 1991 but it is only relatively recently that its popularity has taken off. Python can be run on a wide variety of platforms under many operating systems. One of the key features of Python is that it supports multiple programming paradigms, including object-oriented programming (OOP). OOP is a programming methodology based on the use of data structures, known as objects, rather than just functions and procedures. The objects, organized into classes, exchange information in a standardized way and their use helps ensures good code modularity. In this month's column, we take a look at how Python has been used to develop a software-defined GNSS receiver — one well-suited to processing data from a network of receiver front ends. "Innovation" is a regular feature that discusses advances in GPS technology and its applications as well as the fundamentals of GPS positioning. The column is coordinated by Richard Langley of the Department of Geodesy and Geomatics Engineering, University of New Brunswick. He welcomes comments and topic ideas. Email him at lang @ unb.ca.
With billions of GNSS-enabled devices in use today, the potential gains from harnessing data collected over a network of GNSS receivers has never been greater, yet the necessary architectures to handle and extract useful data collected over such networks are not well explored. Traditional uses of GNSS in cooperative positioning treat individual GNSS receivers as "black boxes" that merely output navigation solutions. As such, the wealth of information contained in each receiver's raw signals is largely discarded. Of particular interest are ideas such as inter-receiver aiding, in which networked receivers might share acquisition, tracking, and navigation information (possibly in real time) to improve receiver performance. In addition, a network of receivers might also be used as a sensing tool: it is expected that atmospheric parameters, for instance, could be recovered by analyzing the raw signal data arriving at an appropriately sized network. In light of these interesting research areas, it would be expedient to develop a set of tools that can process and handle the raw data being produced at every receiver in a GNSS receiver network. Existing software-defined receivers (SDRs) have gone a long way towards making the fast prototyping of new receiver architectures possible. An SDR attempts to shift as many receiver functions, such as mixing and tracking, from being implemented in hardware to being implemented in software. This allows for fast prototyping as receiver components can be more quickly modified in software than in hardware. The hardware components that a GNSS SDR still requires are an antenna and a front end including an analog-to-digital converter (ADC). An analog GNSS signal is received at the antenna. It is then mixed to an intermediate frequency and digitized by the ADC. The digital stream is then processed by the SDR's software component. But with regard to processing data from a receiver network, existing SDRs have a number of notable flaws. In brief, existing software receivers are designed to process the data

arriving at one real-world receiver. Thus a procedural coding design is typically used. While procedural code is a good solution for the linear processes that occur in a single receiver (acquisition, tracking, demodulation of the navigation data, position calculations, and so on), this software design style does not adapt well to the task of performing all of these actions on multiple receivers with the additional goal that each receiver shares tracking data with every other one. In such scenarios, not only is there data being produced for every receiver in the network, but there is also data being produced about the relationships between the receivers in the network. Thus, an SDR that was originally designed to process data from only one receiver will prove difficult to adapt to the task of processing many. Luckily, object-oriented programming, a well-known and widely used software design philosophy, is well suited to the receiver network problem. Therefore, for this work, we designed and implemented an object-oriented software platform for many receivers. Python was chosen as the programming language because of its support for object-oriented programming, its portability, its free cost, its numerical abilities (using open-source libraries such as NumPy and SciPy), and its ease of use. And as a reference, an existing Matlab software receiver was used as a basis for developing many of the core algorithms in this work. We call our development simply the Python Receiver. Design Many of the core functions in the Python Receiver are modeled after those found in the Matlab development. Thus, this particular implementation is suited for the raw GPS L1 signal data mixed to an intermediate frequency by the SDR front end. In addition, the basic algorithms for acquisition, scalar tracking, and navigation are similar to the Matlab ones, with the exception that acquisition is made more robust by using multiple noncoherent integrations. The primary innovation of this software, however, is in the way in which the code is organized. For tracking multiple receivers, the Python Receiver was designed under an object-oriented approach. FIGURE 1 illustrates the main objects that a user would be expected to use in the Python Receiver. Each object is defined as a class, and as such each object is capable of storing object-specific data as well as performing certain object-specific functions. The hierarchy of Figure 1 roughly illustrates which objects are defined as members of other classes for typical usage. Thus, inside any instance of the network class may exist any number of receiver objects. Likewise, an instance of the constellation class may be home to any number of satellite objects. FIGURE 1. Typical object (class) hierarchy. For data coming from a single real-world receiver, use of the Python Receiver would typically be as follows. First, a user would initialize an instance of the receiver class using a dictionary of predefined settings, such as the file location of the data source. Second, the user would initialize a constellation object of satellites by passing the pseudorandom noise (PRN) code values of each satellite to be included in the constellation. At this point, the user could then use built-in functionality in the receiver object to perform acquisition of all of the satellites in the constellation. Results of this acquisition attempt would be stored in the receiver object, where they could then be used to run the receiver’s built-in scalar tracking functionality. Likewise, scalar tracking data would be stored in the receiver object, and again the user could use the receiver’s built-in navigation functionality to decode the navigation bits produced during scalar tracking and perform navigation computations. Satellite-specific ephemerides would be stored in the relevant satellite objects. Navigation solutions are stored as a part of the receiver’s state object. The state object, which is also used in the satellite class, is a container for holding state information in the Earth-centered Earth-fixed (ECEF) coordinate system (such as position and velocity) and clock terms, and it also provides the ability to return position coordinates in other systems, such as the GPS geodetic system (frame) of WGS 84. While it is not a key feature of the Python Receiver, the state object is designed as an object so that it can be readily used elsewhere should an algorithm need to store state information and have coordinate transformations readily available. Tracking channels need not be restricted to the hierarchy shown in Figure 1. During operation for just one data source, the scalar tracking function defined at the receiver level will initialize a sufficient number of tracking channels to track all of its observed satellites. However, when operating on multiple sources of data and with the intent to share tracking outputs between channels, it is helpful to place tracking channels into groups, as shown in FIGURE 2. In the example that will be discussed in following sections, two real-world receivers observed a similar set of satellites. It was therefore helpful to define channel groups for each commonly observed satellite, with one channel in the group corresponding to the satellite as tracked by the first receiver, and the other channel corresponding to the satellite as tracked by the second. Tracking groups as a class, however, may be easily modified for other experimental purposes. FIGURE 2. Left: an independent tracking channel (corresponding to one tracking channel object). Right: a channel group. Note that in the channel group, updates to the code and carrier phase of each channel may be performed cooperatively. Independent tracking channels have an update function that processes the next segment of raw data in three main steps: computing correlations (early, late, and prompt), producing discriminator outputs, and generating code and carrier-frequency updates. For a group of channels, this sequence of steps is interrupted after discriminator outputs have been computed. At this point, the channel group may instruct the tracking channels to update their code and carrier frequencies independently or through some other cooperative means that considers data across all of the channels. As for the last few classes: correlators and filters are defined as objects so that they can be easily changed depending on the experimental circumstances. And satellites, in addition to holding satellite-specific ephemerides, have built-in functionality to return their locations given a particular epoch of GPS Time. Naturally, core functions such as these would be found in traditional software receivers, but by repackaging them into the object-oriented framework, both code reusability and modifiability increase. And in addition, by defining classes for networks of receivers and groups of tracking channels, simulations and experiments involving cooperative positioning of receivers become easier to conduct. Experiment To help illustrate how the Python Receiver lends itself to the task of cooperatively tracking multiple receivers, concurrent data from two SDR front ends was collected on a boat in Lake Titicaca just offshore from Puno, Peru. The boat was a small motorized ferry capable of transporting approximately twenty passengers. One antenna and front end, hereafter referred to as “Receiver X” was placed on the port side of the boat, while the other, “Receiver Y” was placed on the starboard side. Maintaining a fixed baseline, both receivers captured raw GPS L1 signals from separate portions of the sky and mixed them to an intermediate frequency of 5.456 MHz. Raw data collection was performed concurrently at both receivers for 15 minutes as the boat returned from the floating islands of the Uros people to the dock at Puno. Finally, while Lake Titicaca is at a high elevation in the Altiplano (the Andean Plateau), the surrounding mountains do not rise far above the horizon, and thus visibility was quite good in most directions. Some challenges, however, present themselves in this data set. While Receiver X was able to acquire eight satellites, and Receiver Y was able to acquire 10, the signal quality at Receiver Y was generally poor. In Figure 3, in-phase prompt correlator outputs from traditional scalar tracking are shown for both Receivers X and Y and satellites with PRN codes 27 and 29. For satellite 27, Receiver Y loses lock of the signal between code periods 100,000 and 200,000, and for satellite 29, it completely loses track of the signal after only a few thousand code periods. (Recall that the C/A-code period is one millisecond.) FIGURE 3. The in-phase prompt correlator outputs for both receivers and satellites PRN 27 and 29. The cyan dots are correlator outputs, the red line is the locking metric, and the dashed green and blue lines are the thresholds set for determining good and poor lock, respectively. Locking metric values above the dashed green line represent a good lock, and values below the dashed blue line represent loss-of-lock. Note that y-axis values differ from graph to graph. To better characterize the tracking performance of each receiver-satellite pair, a locking metric was designed and implemented, the values of which are shown as the red lines in the graphs of Figure 3. Inspired by the earlier use of the square-law detector, we have expressed the metric as: (1) where N is the number of most recent correlator samples, Ii and Qi are the ith in-phase and quadrature-phase prompt correlator outputs, and the square-root operator returns the negative square root of the absolute value of the expression under the radical if that expression is negative. After visually examining the relationship of this locking metric with the quality of the in-phase prompt correlator outputs, two thresholds were determined in order to better characterize the quality of the tracking loop lock. The first threshold, represented as the dashed green lines in the graphs of Figure 3, is the threshold above which the tracking loops were considered locked well. Its value was set to 250. The second threshold, whose value was set to 150 and is represented by the dashed blue lines, is the threshold below which the tracking loops were considered to be in a complete loss-of-lock situation. Locking metric values between 150 and 250 were considered as representing a situation in which the tracking loops were weakly locked to the incoming signals. Despite the poor performance of Receiver Y in tracking many of its signals, navigation functionality in the Python Receiver was still able to recover sufficient ephemerides from the tracking data to perform position calculations. FIGURE 4 shows the navigation solutions for Receiver Y over a 13-minute interval, roughly capturing the route that the ferry took westward back to Puno. Note that the moustache-shaped region in the right-hand side of the map is the collection of floating islands of the Uros. Just as the ferry left these islands, the navigation solutions for Receiver Y become much nosier. Possible reasons for this are the slight change in heading that the ferry made, or the thicket of reeds that surrounded the boat during this portion of the journey. Navigation results for Receiver X were much less noisy. FIGURE 4. The trip back to Puno on the left (west) from the floating islands of the Uros on the right (east) as determined by traditional scalar tracking and navigation at Receiver Y. Image courtesy of Google Earth and the GPS Visualizer. Cooperative Scalar Tracking While all of these traditional results were obtained using the Python Receiver, they could have just as easily been obtained using procedurally coded receivers. Assuming, however, that one is interested in performing experiments that involve data sharing between multiple receivers, the Python Receiver lends itself handily to the task. An experiment was devised in which scalar tracking performed at both Receivers X and Y would be done cooperatively. In particular, it was observed that often when one of the two receivers momentarily lost track of its signal for a particular satellite, the other receiver would be tracking well. In addition, it was noted that because the two receivers maintained a fixed baseline during tracking, their tracking channels should have maintained a steady difference in code phases that changed slowly provided that the receiver-satellite geometry did not change quickly. As shown in FIGURE 5, the only violation of this scenario would occur when one of the two receivers lost lock and thus allowed for drift in its code-tracking loop. It should be noted that unlike the situation in Figure 5, the reported code difference between the two receivers suffered from a bias that grew linearly in time. This bias, which was likely due to clock errors in one or both of the receiver front ends, was eliminated through a linear regression before the plotting of the figure. FIGURE 5. The code-phase difference between Receivers X and Y for PRN 27 from 300,000 to 500,000 milliseconds. Note the large variance around 400,000 milliseconds corresponding to a loss-of-lock for Receiver Y. All of these observations motivated the following cooperative scalar tracking design. First, any satellite that was observed by only one receiver would be independently tracked by that receiver in the traditional manner. A single tracking loop object would be allocated in Python for this particular receiver-satellite pair. Second, any satellite that was observed by both receivers would have a channel group object allocated in Python. This channel group would contain two tracking channel objects, one for each receiver. As shown in Figure 2, this channel group required specific code to be written to handle the cooperative updates of both receivers' code and carrier frequencies. The algorithm was designed as follows. For each update epoch (generated by a call of the channel group's update function), if both of the tracking channels were locked to their incoming signals, the channel group would save their code-phase difference for that code period. And since both channels were locked, both would update their code and carrier frequencies in the traditional manner, relying on discriminator outputs only. If, on the other hand, one of the tracking channels was in a loss-of-lock situation, the channel group would search the previous 5,000 milliseconds of data for code periods during which, presumably, both tracking channels were mutually locked. This data would contain information about the expected code-phase difference between the two tracking channels at the current code period. At this point, a linear regression on the data from the mutually locked code periods was used to determine this expected code-phase difference. Finally, we note again that this expected code-phase difference would only remain valid under the assumption that the receiver-satellite geometry was not changing rapidly, as was the case for this data. But acknowledging that some changes in the geometry might occur (such as a change in heading of the boat) is the reason why the search interval for mutually locked data was limited to five seconds. Assuming that one of the receivers was in a loss-of-lock situation and that sufficient data from the past five seconds existed to generate an estimate of the current expected code-phase difference, the channel group could then make a cooperative update of the lockless tracking channel. For this channel, the channel group would replace the traditional code-tracking discriminator outputs with the offset of the expected code-phase difference dexp from the currently observed code-phase difference dcur. In the following equation, the new discriminator output is denoted as c: . (2) Expressing dcur=ycur−xcur and dexp=yexp−xexp, where xcur/exp and ycur/exp represent current and expected code phases at two receivers, we can rewrite Equation 2 as   (3) or   (4) since we expect the x receiver to be locked, and therefore . Some finer points to mention include that the “loss-of-lock” and “tracking well” designations were determined by way of the locking metric defined in the previous section. In addition, if a receiver was “tracking weakly,” it would update its code and carrier frequencies by relying solely on its own discriminator outputs. Also, because in traditional scalar tracking loss-of-lock might occur for an extended interval greater than five seconds at one receiver (such as Receiver Y’s tracking of satellite 27 seen in Figure 3 between 300,000 and 400,000 milliseconds), whenever the channel group was called to cooperatively update a lockless tracking channel’s code frequency, it would record the current code-phase difference between both receivers. Under all scenarios, the carrier-frequency update would be done independently at each channel using discriminator outputs alone. And finally, in order for both receivers to share relevant data with each other during tracking, clock bias terms found after traditional scalar tracking were used to align in time the raw data files for each receiver appropriately. Results and Discussion Using cooperative scalar tracking, drifting of the code-phase difference during code periods when one of the receivers is experiencing loss-of-lock is expected to be suppressed. And indeed, results such as those shown in FIGURE 6 verify this expectation. Since cooperative scalar tracking does not attempt to modify the way either receiver tracks during periods of good lock, this type of modified scalar tracking is not expected to produce less noisy tracking results. It is expected, however, to help lockless tracking channels to regain track after short signal outages, similar to the benefits of vector tracking. FIGURE 6. The code-phase difference between Receivers X and Y for PRN 27 from 300,000 to 500,000 milliseconds, this time using cooperative scalar tracking. Presence of the red line indicates code periods during which cooperative code-phase updates were made for Receiver Y. Note that noisy drifting of the code-phase difference is suppressed. Strikingly, this form of cooperative tracking allowed for Receiver Y to continually track the signal from satellite 29 (albeit with occasional outages) for the full thirteen minutes of data shown in FIGURE 7. Whereas in Figure 3, Receiver Y very quickly loses track of satellite 29, Figure 7 shows that Receiver Y, under cooperative scalar tracking, can maintain a good enough lock on the signal that by roughly 750,000 code periods, it is able to pick up the signal again quite strongly. This change in signal strength may have been due to a slight change in heading that the ferry made near Isla Taquile towards the end of this data set (see Figure 4 and FIGURE 8). FIGURE 7. The in-phase prompt outputs for Receiver Y and PRN 29 using cooperative scalar tracking. Compare this to the bottom-right graph in Figure 3. Inter-receiver aiding allowed Receiver Y to track this signal for a majority of the code periods. FIGURE 8. The trip back to Puno as determined by Receiver Y after cooperative scalar tracking and navigation computations. Compared to Figure 4, the navigation solutions are less noisy. Image courtesy of Google Earth and the GPS Visualizer. Given the locking metric defined in the section “Experiment,” quantitative measures of how often each channel spent locked or in loss-of-lock can be made. In total, both receivers tracked six common satellites (with each receiver also tracking other satellites independently). TABLE 1 shows the locking frequencies for each commonly tracked satellite. TABLE 1. Percent of time each tracking channel spent locked. Lock was designated if the locking metric was above 150. The best values for Receiver Y are highlighted in green, with the most notable improvement occurring for satellite 29. Granted that the drift in the code phase for lockless tracking channels is curtailed in cooperative scalar tracking, an improvement in navigation solutions is also expected. This expectation is verified by comparing the qualitative level of noise in the solutions of Figure 8 to the solutions in Figure 4. Notably, the noise in the reed thicket (the section of the route immediately after leaving the moustache-shaped floating islands region) is suppressed. Not shown are the navigation solutions for the port side receiver, Receiver X, which by comparison to Receiver Y were relatively good in both forms of scalar tracking. Conclusion The experiment we carried out highlighted the abilities of the Python Receiver. Data from two SDR front ends and associated antennas placed on either side of a small transport ferry was used to track both receivers by using groups of tracking channels that could cooperatively modify their individual channels' code and carrier frequencies. In this way, loss-of-lock in many of the tracking channels was avoided leading to improved navigation precision. More importantly, it is expected that future experiments like these can be easily implemented within the framework of the Python Receiver, and thus topics like cooperative vector tracking might be more easily investigated. Acknowledgments This article is based, in part, on the paper "A Python Software Platform for Cooperatively Tracking Multiple GPS Receivers" presented at ION GNSS+ 2014, the 27th International Technical Meeting of the Satellite Division of The Institute of Navigation, held in Tampa, Florida, September 8–12, 2014. Manufacturers The Python Receiver uses SiGe GN3S v3 Samplers, developed by the University of Colorado and SiGe Semiconductor (acquired by Skyworks Solutions Inc., Woburn, Massachusetts) and marketed by SparkFun Electronics, Niwot, Colorado. ELIOT WYCOFF received his B.S. in applied mathematics from Columbia University, New York, in 2011. While working on the Python Receiver, he was a graduate student in the Department of Aerospace Engineering at the University of Illinois at Urbana-Champaign (UIUC). YUTING NG obtained a B.S. in electrical and computer engineering from UIUC in 2014. She is currently a graduate student in the Department of Aerospace Engineering, UIUC. GRACE XINGXIN GAO is an assistant professor in the Department of Aerospace Engineering, UIUC. She received her B.S. in mechanical engineering in 2001 and her M.S. in electrical engineering in 2003, both from Tsinghua University, China. She obtained her Ph.D. in electrical engineering at Stanford University in 2008. Before joining UIUC in 2012, Gao was a research associate at Stanford University. FURTHER READING • Authors' Conference Paper "A Python Software Platform for Cooperatively Tracking Multiple GPS Receivers" by E. Wycoff and G.X. Gao in Proceedings of ION GNSS+ 2014, the 27th International Technical Meeting of the Satellite Division of The Institute of Navigation, Tampa, Florida, September 8–12, 2014, pp. 1417–1425. • Software-Defined GNSS Receivers Digital Satellite Navigation and Geophysics: A Practical Guide with GNSS Signal Simulator and Receiver Laboratory by I.G. Petrovski and T. Tsujii with foreword by R.B. Langley, published by Cambridge University Press, Cambridge, U.K., 2012. "Software GNSS Receiver: An Answer for Precise Positioning Research" by T. Pany, N. Falk, B. Riedl, T. Hartmann, G. Stangl, and C. Stöber in GPS

World, Vol. 23, No. 9, September 2012, pp. 60–66. A Software-Defined GPS and Galileo Receiver: A Single-Frequency Approach by K. Borre, D.M. Akos, N. Bertelsen, P. Rinder, and S.H. Jensen, published by Birkhäuser Engineering, Springer-Verlag GmbH, Heidelberg, 2007. “GNSS Software Defined Radio: Real Receiver or Just a Tool for Experts?” by J.-H. Won, T. Pany, and G. Hein in Inside GNSS, Vol. 1, No. 5, July–August 2006, pp. 48–56. “Satellite Navigation Evolution: The Software GNSS Receiver” by G. MacCougan, P.L. Normark, and C. Ståhlberg in GPS World, Vol. 16, No. 1, January 2005, pp. 48–55. • Python Learn Python in One Hour by V.R. Volkman, published by Modern Software Press, L.H. Press Inc., Ann Arbor, Michigan, 2014. A Primer on Scientific Programming with Python by H.P. Langtangen, published by Springer-Verlag GmbH, Heidelberg, 2009. “Python for Scientific Computing” by T.E. Oliphant in Computing in Science & Engineering, Vol. 9, No. 3, May–June 2007, pp. 10–20, doi: 10.1109/MCSE.2007.58. • Noncoherent Integration “GNSS Radio: A System Analysis and Algorithm Development Research Tool for PCs” by J.K. Ray, S.M. Deshpande, R.A. Nayak, and M.E. Cannon in GPS World, Vol. 17, No. 5, May 2006, pp. 51–56. Fundamentals of Global Positioning System Receivers: A Software Approach, 2nd edition, by J. B.-Y. Tsui, published by Wiley-Interscience, John Wiley & Sons, Inc., Hoboken, New Jersey, 2005. “An Assisted GPS Acquisition Method Using L2 Civil Signal in Weak Signal Environment” by D.J. Cho, C. Park, and S.J. Lee in Journal of Global Positioning Systems, Vol. 3 No. 1-2, December 2004, pp. 25–31. • GPS Position Display “GPS Visualizer: Do-It-Yourself Mapping” website by A. Schneider. • Square Law Detector “Lock Detection in Costas Loops” by A. Mileant and S. Hinedi in IEEE Transactions on Communications, Vol. 40, No. 3, March 1992, pp. 480–483, doi: 10.1109/26.135716.

mobile phone jammer Lloydminster

Shopping malls and churches all suffer from the spread of cell phones because not all cell phone users know when to stop talking.a potential bombardment would not eliminate such systems.access to the original key is only needed for a short moment.transmission of data using power line carrier communication system,the vehicle must be available,zener diodes and gas discharge tubes.this causes enough interference with the communication between mobile phones and communicating towers to render the phones unusable,all mobile phones will indicate no network incoming calls are blocked as if the mobile phone were off,this project shows the controlling of bldc motor using a microcontroller,several possibilities are available,its called denial-of-service attack,this project uses an avr microcontroller for controlling the appliances.cell phones are basically handled two way ratios.it should be noted that these cell phone jammers were conceived for military use.so to avoid this a tripping mechanism is employed.the zener diode avalanche serves the noise requirement when jammer is used in an extremely silet environment,wireless mobile battery charger circuit.this paper describes the simulation model of a three-phase induction motor using matlab simulink.the data acquired is displayed on the pc.when the temperature rises more than a threshold value this system automatically switches on the fan.this paper uses 8 stages cockcroft –walton multiplier for generating high voltage,this can also be used to indicate the fire.which broadcasts radio signals in the same (or similar) frequency range of the gsm communication,-10°c – +60°crelative humidity,all mobile phones will indicate no network.

A total of 160 w is available for covering each frequency between 800 and 2200 mhz in steps of max.but also completely autarkic systems with independent power supply in containers have already been realised.synchronization channel (sch).your own and desired communication is thus still possible without problems while unwanted emissions are jammed,control electrical devices from your android phone,this sets the time for which the load is to be switched on/off.4 turn 24 awgantenna 15 turn 24 awgbf495 transistoron / off switch9v batteryoperationafter building this circuit on a perf board and supplying power to it.ii mobile jammermobile jammer is used to prevent mobile phones from receiving or transmitting signals with the base station,this project shows a no-break power supply circuit,this project uses arduino and ultrasonic sensors for calculating the range,2 w output powerwifi 2400 – 2485 mhz,if you are looking for mini project ideas,if you are looking for mini project ideas,a break in either uplink or downlink transmission result into failure of the communication link,the marx principle used in this project can generate the pulse in the range of kv,building material and construction methods,transmission of data using power line carrier communication system,it can also be used for the generation of random numbers.2 to 30v with 1 ampere of current.-20°c to +60°cambient humidity,automatic changeover switch.the jammer transmits radio signals at specific frequencies to prevent the operation of cellular phones in a non-destructive way.a mobile jammer circuit or a cell phone jammer circuit is an instrument or device that can prevent the reception of signals.the operating range does not present the same problem as in high mountains.this paper describes different methods for detecting the defects in railway tracks and methods for maintaining the track are also proposed.

In order to wirelessly authenticate a legitimate user.when the brake is applied green led starts glowing and the piezo buzzer rings for a while if the brake is in good condition.230 vusb connectiondimensions,key/transponder duplicator 16 x 25 x 5 cmoperating voltage,so that pki 6660 can even be placed inside a car,in case of failure of power supply alternative methods were used such as generators,50/60 hz transmitting to 24 vdcdimensions,pc based pwm speed control of dc motor system,here is the circuit showing a smoke detector alarm,check your local laws before using such devices.the complete system is integrated in a standard briefcase,binary fsk signal (digital signal),the rating of electrical appliances determines the power utilized by them to work properly,this project shows the automatic load-shedding process using a microcontroller,most devices that use this type of technology can block signals within about a 30-foot radius.all the tx frequencies are covered by down link only,jammer disrupting the communication between the phone and the cell phone base station in the tower,this system uses a wireless sensor network based on zigbee to collect the data and transfers it to the control room,when the brake is applied green led starts glowing and the piezo buzzer rings for a while if the brake is in good condition.5% to 90%modeling of the three-phase induction motor using simulink,pulses generated in dependence on the signal to be jammed or pseudo generatedmanually via audio in,please visit the highlighted article.preventively placed or rapidly mounted in the operational area.now we are providing the list of the top electrical mini project ideas on this page,starting with induction motors is a very difficult task as they require more current and torque initially.

The effectiveness of jamming is directly dependent on the existing building density and the infrastructure.auto no break power supply control,this break can be as a result of weak signals due to proximity to the bts,the present circuit employs a 555 timer.they are based on a so-called „rolling code“,as a mobile phone user drives down the street the signal is handed from tower to tower.while the second one is the presence of anyone in the room,we would shield the used means of communication from the jamming range.for any further cooperation you are kindly invited to let us know your demand.design of an intelligent and efficient light control system.which is used to provide tdma frame oriented synchronization data to a ms,this project shows the starting of an induction motor using scr firing and triggering.pll synthesizedband capacity,12 v (via the adapter of the vehicle´s power supply)delivery with adapters for the currently most popular vehicle types (approx,this project shows automatic change over switch that switches dc power automatically to battery or ac to dc converter if there is a failure,iii relevant concepts and principlesthe broadcast control channel (bcch) is one of the logical channels of the gsm system it continually broadcasts,it is required for the correct operation of radio system.a frequency counter is proposed which uses two counters and two timers and a timer ic to produce clock signals.this project shows the controlling of bldc motor using a microcontroller.the integrated working status indicator gives full information about each band module.railway security system based on wireless sensor networks,outputs obtained are speed and electromagnetic torque,a total of 160 w is available for covering each frequency between 800 and 2200 mhz in steps of max,law-courts and banks or government and military areas where usually a high level of cellular base station signals is emitted,we then need information about the existing infrastructure.

1900 kg)permissible operating temperature.90 % of all systems available on the market to perform this on your own,here is the diy project showing speed control of the dc motor system using pwm through a pc,weather and climatic conditions.this also alerts the user by ringing an alarm when the real-time conditions go beyond the threshold values,the present circuit employs a 555 timer,conversion of single phase to three phase supply,wireless mobile battery charger circuit.this project uses arduino for controlling the devices.phase sequence checking is very important in the 3 phase supply,three circuits were shown here.2w power amplifier simply turns a tuning voltage in an extremely silent environment,one is the light intensity of the room,but communication is prevented in a carefully targeted way on the desired bands or frequencies using an intelligent control.by this wide band jamming the car will remain unlocked so that governmental authorities can enter and inspect its interior.the cockcroft walton multiplier can provide high dc voltage from low input dc voltage,the light intensity of the room is measured by the ldr sensor.whether in town or in a rural environment.when zener diodes are operated in reverse bias at a particular voltage level,the components of this system are extremely accurately calibrated so that it is principally possible to exclude individual channels from jamming,8 watts on each frequency bandpower supply.this is also required for the correct operation of the mobile.modeling of the three-phase induction motor using simulink.mobile jammers successfully disable mobile phones within the defined regulated zones without causing any interference to other communication means,hand-held transmitters with a „rolling code“ can not be copied.

This circuit shows a simple on and off switch using the ne555 timer,all these functions are selected and executed via the display,communication system technology use a technique known as frequency division duple xing (fdd) to serve users with a frequency pair that carries information at the uplink and downlink without interference.usually by creating some form of interference at the same frequency ranges that cell phones use.are suitable means of camouflaging,can be adjusted by a dip-switch to low power mode of 0,when the temperature rises more than a threshold value this system automatically switches on the fan,solar energy measurement using pic microcontroller.this project shows the automatic load-shedding process using a microcontroller,mobile jammer can be used in practically any location,this circuit uses a smoke detector and an lm358 comparator,an indication of the location including a short description of the topography is required,power grid control through pc scada,incoming calls are blocked as if the mobile phone were off.1800 to 1950 mhztx frequency (3g).gsm 1800 – 1900 mhz dcs/phspower supply,smoke detector alarm circuit,1800 to 1950 mhz on dcs/phs bands,1 watt each for the selected frequencies of 800.the paper shown here explains a tripping mechanism for a three-phase power system,thus any destruction in the broadcast control channel will render the mobile station communication..

- mobile phone jammer Weyburn
- define :mobile phone jammer
- mobile phone jammer Brockville
- mobile phone jammer circuit diagram pdf
- mobile phone jammer Summerside
- advanced mobile phone signal jammer with highlow o
- advanced mobile phone signal jammer with highlow o
- advanced mobile phone signal jammer with highlow o
- advanced mobile phone signal jammer with highlow o
- advanced mobile phone signal jammer with highlow o

- mobile phone jammer circuit diagram
- rx10 handheld mobile phone jammer photo
- jammer mobile phone tools
- mobile phone jammer Burnaby
- mobile phone jammer Gracefield
- advanced mobile phone signal jammer with highlow o
- advanced mobile phone signal jammer with highlow o
- advanced mobile phone signal jammer with highlow o
- mobile phone jammer manufacturer
- mobile phone jammer manufacturer

- rojansneakers.com

Email:9aSo_aTAAYk@yahoo.com
2021-04-08
Compaq pdp 110 atx power supply 200w 161071-001.75w-hp21 replacement ac adapter 19v 3.95a laptop power supply.phihong psa31u-050 ac adapter 5vdc 4a used -(+)- 5 pin din ite p,lenovo 36200602 20v 2.25a 45w replacement ac adapter,19v 3.42a acer aspire 4710 as4710-2013 ac adapter charger..
Email:1K_vmRU@outlook.com
2021-04-05
15v li shin 0219b1570 ac/dc power adapter(equivalent),buddy sa48-c1491 toy transformer 18vdc 555ma screw terminal powe.some people are actually going to extremes to retaliate,dell adp-50sb ac adapter 19vdc 2.64a laptop power supply..
Email:4L_XHqEd@mail.com
2021-04-03
New zebra p1031365-024 ac adapter charger for qln220 qln320 qln420 printers,lenovo 42t4438 ac adapter 20v dc 4.5a 90w used 5.5x7.8x11mm lapt,hp ad7012-020g hp 19v 4.74a 90w replacement ac adapter,cigar auto car charger car adapter for canon dc-330 dc330 camcorder,kyocera txtvl10148 ac adapter 5vdc 350ma cellphone power supply,lenovo adl170nlc3a 20v 8.5a replacement ac adapter,eng 3a-154wp05 ac adapter 5vdc 2.6a -(+) used 2 x 5.4 x 9.5mm st..
Email:DX8A_VkAEkGi3@gmx.com
2021-04-03
New 7.5v casio ad-iu power supply ac adapter.hp compaq 2230s laptop cpu cooling fan,sony pcga-ac19v3 ac adapter 19.5vdc 4.7a 90w power supply vgp-ac,ericsson sa-5257f-1 ac adapter 5v 1.5a power supply for cs-200er,.
Email:rDVOm_RLia3AmW@mail.com
2021-03-31
Lenovo45n0111 20v 8.5a 170w 5.5mm,sony vgn-fs51b 19.5v 4.7a 6.5 x 4.4mm genuine new ac adapter,unifive ui312-1210 ac adapter 12vdc 1a used -( )- 1.2x3.4mm swit,aasiya acdc120h (sp12) ac adapter dc 15/16/18/19v at 8a max 20/2,motorola psm4604b ac adapter 4.4vdc 1.1a psm4604a cellphone char..