SOFTWARE-DEFINED WIRELESS NETWORK FOR REAL-TIME

SENSING

An Undergraduate Research Scholars Thesis

by

AUSTIN M. KEITH

Submitted to the LAUNCH: Undergraduate Research office at Texas A&M University in partial fulfillment of requirements for the designation as an

UNDERGRADUATE RESEARCH SCHOLAR

Approved by Faculty Research Advisor:

Dr. I-Hong Hou

May 2022

Major:

Electrical Engineering

Copyright © 2022. Austin M. Keith.

RESEARCH COMPLIANCE CERTIFICATION

Research activities involving the use of human subjects, vertebrate animals, and/or biohazards must be reviewed and approved by the appropriate Texas A&M University regulatory research committee (i.e., IRB, IACUC, IBC) before the activity can commence. This requirement applies to activities conducted at Texas A&M and to activities conducted at non-Texas A&M facilities or institutions. In both cases, students are responsible for working with the relevant Texas A&M research compliance program to ensure and document that all Texas A&M compliance obligations are met before the study begins.

I, Austin M. Keith, certify that all research compliance requirements related to this Undergraduate Research Scholars thesis have been addressed with my Research Faculty Advisor prior to the collection of any data used in this final thesis submission.

This project did not require approval from the Texas A&M University Research Compliance & Biosafety office.

TABLE OF CONTENTS

Page	

ABSTRACT1								
DEDICATION								
ACKNOWLEDGEMENTS								
NOMENCLATURE								
CHAPTERS								
1. INTRODUCTION								
1.1Traditional Networking61.2Software-Defined Networking81.3Software-Defined Radio101.4Complete System12								
2. METHODS								
2.1SDN Construction								
3. RESULTS								
 3.1 Operation of the Complete System								
4. CONCLUSION								
 4.1 Effectiveness of SDN-SDR Network								
REFERENCES								
APPENDIX A: PYTHON SOFTWARE WRITTEN FOR COMMUNICATION								

ABSTRACT

Software-Defined Wireless Network for Real-Time Sensing

Austin M. Keith Department of Electrical & Computer Engineering Texas A&M University

Research Faculty Advisor: Dr. I-Hong Hou Department of Electrical & Computer Engineering Texas A&M University

Traditionally in the field of electronics, hardware is designed, developed, and improved on in various methods, whether it be increased storage capabilities or smaller models. Software applications lagged because of the hardware requirements to operate software, but increasingly, software tools are replacing technology that relied heavily on hardware components where applicable because of the abilities to both modify the technology easily and to consolidate tasks in an automated fashion. This research focuses on the networking space and aims to replace hardware architecture with software, as well as write algorithms to intelligently allocate incoming data.

To orchestrate this architecture and the algorithms, the modern tools of software-defined networking and software-defined radios were combined. This created a network capable of transmitting packets over-the-air, with the network itself having separated the data plane and control plane in the software-defined networking standard. The control plane is written entirely in software, allowing modifications to be made across the whole system relatively simply. In this research, two software-defined radios were used to represent a base station and a field multi-

sensor collector respectively. The field sensor transmits real sensor data from a web database that represents readings of the resistance of a gas over time from fourteen sensors. The base station radio can only receive a single packet at a time from the secondary radio due to bandwidth constraints, and so, using a software-defined controller, the various scheduling policies are compared to develop the most efficient means of processing the individual data packets.

The final algorithm started from basic round-robin before evolving into weighted roundrobin, with measurable results in terms of root-mean-square error values for each sensor and one for the total transmission period. The weighted round-robin was upgraded a step further to have real-time weight updates at regular intervals based on the accuracy of prediction for the next value in the sequence, per sensor. The contrast between the three stages of development for the round-robin algorithms is plain to see, with steady improvement between basic round-robin and weighted round-robin, and drastic improvement between weighted round-robin and the smart algorithm. The results from the research project yield a final draft of communication between the software-defined radios that produced an effective and efficient manner of software-defined networking.

DEDICATION

To my family, for supporting me continuously. To my fiancé, for always being there. To my friends, for keeping me focused.

ACKNOWLEDGEMENTS

Contributors

I would like to thank my faculty advisor, Dr. I-Hong Hou, and my graduate student colleagues, Siqi Fan and Khaled Nakhleh, for their guidance and support throughout the course of this research.

Thanks also go to my friends and colleagues and the department faculty and staff for making my time at Texas A&M University a great experience.

Finally, thanks to my parents for their support and encouragement and to my fiancé for her patience and love.

The B210 software-defined radios used for over-the-air communication were provided by the Wright Brothers Institute and the Air Force Research Laboratory for participating in the Beyond 5G University Challenge for 2021-2022.

All other work conducted for the thesis was completed by the student independently.

Funding Sources

No sources for funding were used for the duration of this research.

NOMENCLATURE

- USRP Universal Software Radio Peripheral
- SDN Software-Defined Network
- $SDR-Software\text{-}Defined\ Radio$
- GUI Graphical User Interface
- RF Radio Frequency
- GMSK Gaussian Minimum Shift Keying
- GHz-Gigahertz
- RMSE-Root-Mean-Square-Error
- UDP User Datagram Protocol
- WRR Weighted Round-Robin

1. INTRODUCTION

The goal of this project is to have a software-defined network that takes advantage of two software-defined radios, one working as a base station and the other representing a series of field sensors, to transmit a variety of sensor data over-the-air. Because of bandwidth constraints, only a single data packet can transmit through the radios at a single instant of time. Algorithms written for the controller of the network allocate the incoming sensor data efficiently to reduce the time it takes to process the incoming data. Because much of the operation of the entire system can be modified through the controller software, the network can switch between various scheduling priorities to demonstrate the differences in efficiency between them.

1.1 Traditional Networking

To begin the understanding of this research project, one must understand how a traditional network works as well as the issues surrounding it. Networking itself is simply a connection of two or more devices that allows them to share information with each other. The number of connections and geographical range of the connections determines the categorization of the network, but there are common elements between all traditional network types. Each are built on connections between computers that are created with the use of switches and routers, as shown in Figure 1.1 below. Switches ensure that data packets are communicated and sent properly within a single network, while routers ensure that data packets are correctly transmitted between networks themselves. This creates a hierarchy of networks, routers, and switches that are highly dependent on physical hardware for the establishment of the network and for future modifications.

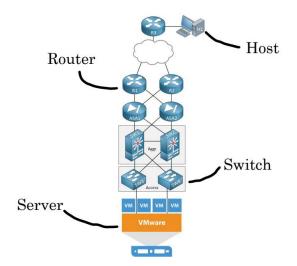


Figure 1.1: Network Components

An example of such a network on the larger side of the scale is the Internet, which is a giant web of interconnected networks so that any device can send information to any other device. Other examples include university campus networks, a network of computers in an office building, and a network of gamers playing online together. Each of these examples is highly dependent on hardware, limiting its flexibility and increasing its complexity as the number of users join.

1.1.1 Problems with Traditional Networking

Some of the issues associated with this style of network building and amendment have been hinted at. Most of the current problems with traditional networking can be categorized under either an issue of complexity or an issue of inefficiency.

The immediate problem with traditional networking is that the number of users has grown at a tremendous rate in the last few decades. Both the number of people that own devices has increased as well as the number of devices that a single individual owns. Regardless of the networks that these devices are connected to, the overall complexity of networking has increased dramatically because of the number of connections that needs to be maintained has increased dramatically. Each additional connection requires the use of hardware to either make physical connections (with a cable) or virtual connections (with Wi-Fi, for example). This creates an enormous number of switches, routers, and other network architecture hardware that must be independently maintained to ensure the network operates as intended. Additionally, any changes in the way that devices are connected, including emergency changes if a critical part of the network fails, are slow to happen and difficult to execute. Network engineers must be welltrained and staffed at many companies just to accomplish this difficult work.

The inefficiencies of traditional networking may be obvious with many disciplines of technology turning towards software, even in fields that were dominated by hardware in the past. The limitations on flexibility with physical hardware in the environment of increased complexity create problems in areas of high traffic in a network in a live setting, and the potential for network connections to adapt in a highly efficient manner to the information that they communicate is quite high. However, even in an offline setting, changes to a network are limited because multiple parts of the network must be individually modified. The problem of multiple network engineers changing different switches and routers without communicating and causing conflicts with the network connections also occurs.

1.2 Software-Defined Networking

The newer type of networking that is starting to be employed is known as softwaredefined networking, or SDN. SDN differs largely from the traditional network due to an abstract layer of computer architecture known as the control plane which is separated from the data plane in SDN. Before, the data plane and control plane were paired together, meaning in practicality that each fixed piece of hardware in the network that is responsible for sending packets was

independently operated. With SDN, the control plane is separated into a centralized unit known as the controller, and the devices responsible for communication are directed by the controller, as shown in Figure 1.2. In this manner, a proper analogy is that traditional network architecture is orchestrated like a pack of wolves, where each individual animal makes decisions so that the pack can survive. A similar analogy can be made for SDNs which operate like a beehive, with individual worker units that have a hive mind and adapt to new situations collectively.

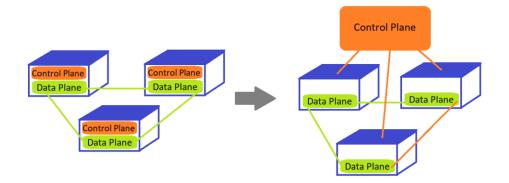


Figure 1.2: Control Plane Separation

Because the SDN controller is programmable and runs the network, separating the control plane and data plane has huge benefits to networking and addresses many of the core issues defined with traditional networking.

1.2.1 Benefits with SDN

There are many benefits associated with switching from a traditional network to a software-defined network. The first and most obvious benefit is that network operation is controlled by a single, programmable unit. This means that, rather than a network engineering modifying switches and packets at an individual level, a single network engineer can write code for a controller that modifies the entire system. This greatly reduces the complexity of managing a given network. Adjustments are automatically made within the network to adapt to the new

communication methods written in the controller software. The increased level of flexibility this provides cannot be understated. Adding multiple devices to a network becomes much simpler as the controller adjusts virtual switches and routers to accommodate new users. Not only can a single network be changed drastically in a short period of time, but an SDN can be repurposed for multiple different functions as well. Established virtual networks that are no longer needed can be modified to serve a new function without the steps necessary in traditional networking to rebuild the architecture.

Additionally, maintenance of a network using SDN technology is much simpler. The controller operates by changing the connection lines and general network architecture to address the conditions set by an engineer within the software. This allows automatic readjustments in connections between devices when errors occur. Network failures at critical traffic points are no longer a major concern while the network is manually repaired as done with traditional networking; instead, the SDN establishes new connections automatically to ensure operation continues as intended. While this implies benefits that address the complexity and flexibility issues described under the traditional networking infrastructure, it also improves the security of networks. Point-to-point connections that are attacked by malicious users can adapt with a new secure channel of communication once the network controller recognizes that an attack is happening.

1.3 Software-Defined Radio

In a similar line of thought, radios can have a huge increase in flexibility and a reduction in maintenance cost and complexity by moving some part of the physical functionality into the software domain. This precisely describes software-defined radios (SDRs), where some aspect of the physical elements of a traditional radio are controlled by programmable software. Some

physical components of a radio are necessary for transmission and signal processing, but the general idea of separating the control plane into software applies here just as done with SDN. There are a great variety of types of SDRs as their share of the market increased over the last few decades, and there are a multitude of benefits that come from using an SDR over a traditional-style radio. In Figure 1.3, the two radios that are used for the duration of this project are shown.



Figure 1.3: Two Software-Defined Radios (B210s)

1.3.1 Benefits of SDR

Several of the benefits associated with SDN over traditional networking are reflected in the benefits of choosing SDR over traditional radios. Modification of the purpose of the SDR is simple and only requires a change in the code that controls its operation. Similarly, maintenance of the radio is much less complicated when, instead of physically tampering with the device, new code can be uploaded that addresses the problem. Both simplifications massively reduce the cost of using the technology and provide greater flexibility than previously offered.

The added flexibility that comes with the programmable functions of the SDR is another benefit. A group of radios can be linked together and controlled through their software to do a variety of tasks that were unavailable previously. Commands or data packets that are sent overthe-air using SDR technology allows devices in a network or similar infrastructure to be modified remotely. Radio frequency processing can be adjusted easily, allowing communication channels to rapidly change. This comes with increased security benefits as well, with emphasis on application within military technology due to the large use of traditional radios.

1.4 Complete System

The task of this project is to combine the new technologies and applications of SDR and SDN into a single system, and then to write algorithms that receive data packets and allocates them efficiently. This system is intended to be a software-defined network containing two SDRs that communicate to each other. Specifically, the controller for the SDN sends commands to a virtual switch by sending the command to the first SDR, which then gets transmitted over-the-air to the second SDR, and finally, the second SDR sends the command to the switch. The switch responds appropriately to the command, most of which were written previously by graduate students in this research field. In Figure 1.4, a block diagram of the complete architecture for the system is presented.

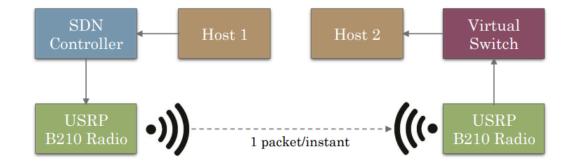


Figure 1.4: Complete Architecture of System

Some of the commands that are sent through this system change the scheduling priority of the SDN. Traditional scheduling methods, such as round-robin, are included, along with the other algorithms written for the purpose of this project.

2. METHODS

The approach to this research project is best described in two categories: the construction of the SDN and the development of the communication between the base station and field sensor SDRs. The first semester was primarily focused on the early work for implementing the SDN on the hosts and the research components of the project, while the winter break and second semester were instead focused on creating the block diagrams and Python files for the radio connections, writing the scheduling priority algorithms to allocate the data appropriately, and completing the infrastructure of the project.

2.1 SDN Construction

Much of the work involved in the initialization of the SDN was about reconstructing the basics from the work of previous graduate students using the open-source information from Ryu and OpenFlow (developed basic controller for network and methods for configuring settings). The terminal commands for constructing the first edition of the SDN were included in some documentation from the previous semesters, and using that information along with the public instructions, the SDN was first created on a single host running multiple virtual machines. The controller terminal, switch terminal, and a connection terminal were all running independently, and communication was confirmed with fake data being sent to a GUI that was included in the work done by the previous graduate students. Figures 2.1 and 2.2 show the early user interaction with the terminal and GUI when establishing the SDN.

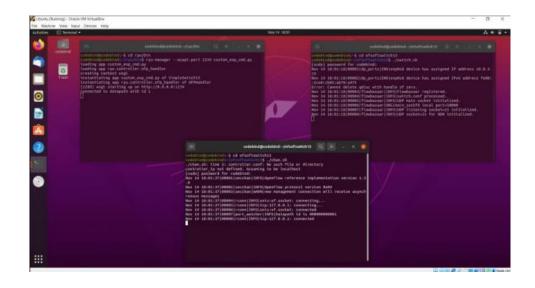


Figure 2.1: Terminal Interface for Switch and Controller



Figure 2.2: GUI Interface for SDN Controller

2.1.1 UDP Connection from USRPs to Localhosts

The original plan for the next step in SDN research was to create a UDP connection between the respective localhost and radio connected to the host. This was not known to be possible, so the alternative plan of improving the weighted round-robin algorithm with a smart algorithm that updates the weights of sensors in real-time was created. In the end, there was no solution found on addressing this UDP connection, so the backup plan was implemented, and a smart algorithm was created. The SDN aspect of this project, because of this change, was less significant and primarily was relegated to research purposes.

2.2 SDR Communication

The construction of the architecture that allows communication between the two SDRs originally used the GNU Radio Companion software tool to construct block diagrams using the pre-built library meant for SDRs and other communication equipment. From there, the generated Python files from the block diagrams within GNU Radio Companion were modified by hand to develop the specific needs regarding this research. Each host device had its own radio, software, and Python files that were created collectively before individual modifications pertaining to the base station or the field sensors respectively were implemented.

2.2.1 One-Way Communication

The first block diagrams created were to establish some form of basic communication between the two radios. After researching the toolset within GNU Radio Companion and understanding the capabilities of the included functions, the transmission and reception files were developed on each respective radio's host device. The method of communication uses GMSK modulation on the transmit side, and GMSK demodulation on the receive side. The two B210 SDRs are both using a channel frequency of 1 GHz and a sample rate of 400k samples per second. These were chosen with the device characteristics and the nature of the project in mind. The two block diagrams that represent the files created to first establish one-way communication are displayed below in Figures 2.3 and 2.4.

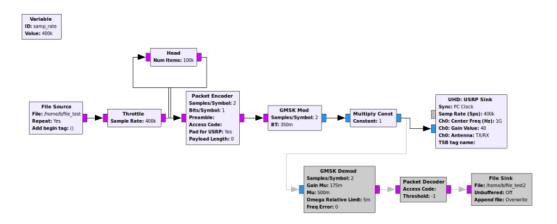


Figure 2.3: Transmission Block Diagram for One-Way Communication

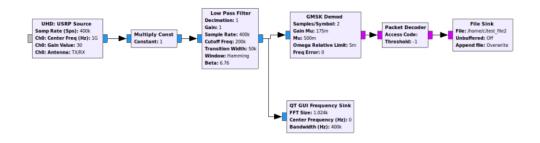


Figure 2.4: Reception Block Diagram for One-Way Communication

The actual data that was being sent from the first SDR to the second SDR came from a local text file on the transmission host with a series of fake data points created for the purpose of establishing one-way communication. The data is read from the text file, encoded into packets, modulated, and sent through the B210s before the process is reversed on the other side and the data is stored onto a new text file on the receiving host. This method of using a text file to send fake data is repeated until the SDN commands are implemented into the SDR's communication.

Running both files on each host at the same time resulted in the file being correctly transferred to the other side, with the only issue being a glitch that is built into GNU Radio

Companion that requires the message to repeat continuously. This is addressed in a future upgrade to the SDR communications.

2.2.2 Two-Way Communication

The first upgrade needed for the communication between the two SDRs is to establish two-way communication, or the ability for both radios to transmit and receive information. Once again, additional block diagrams are built in GNU Radio Companion for these modifications. Fortunately, much of the content of the blocks is the same as when establishing one-way communication, but the specific settings regarding the antennas being used and the timing when either actively sending or actively listening to a signal are changed to meet the new requirements. The block diagrams used for this point of the project are shown in Figures 2.5 and 2.6 below.

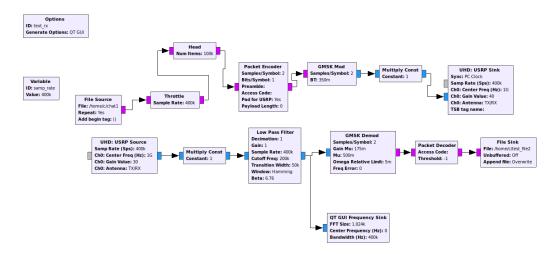


Figure 2.5: Two-Way Communication on Transmit Side

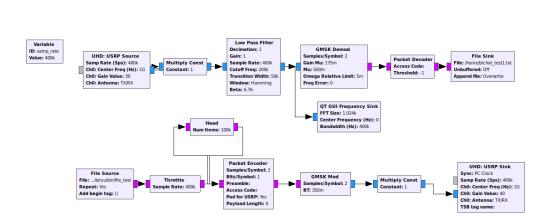


Figure 2.6: Two-Way Communication on Receive Side

After constructing the block diagrams for the radios, the generated Python files are modified to match the style of communication needed for this project. This means that timings and order are introduced to the devices; each respective device needs to be able to understand that it can only listen while it is not talking, and that while it is talking, it is not listening. These changes allow basic conversation between the SDRs to take place without interruption or other complications to the dialogue.

Once the Python files have been adjusted, the radios can send information back and forth to each other. The text file filled with fake data referenced previously for testing functionality of the radio communications is once again used here.

2.2.3 Transfer from Multiple Text Files

Options ID: text_tx Generate Options: QT GU

The next step in terms of communication between the radios is to add the capability to send from multiple options of text files on the field sensor side of the SDRs. In this model, the field sensor SDR has three text files, each filled with random data and identifier numbers for each data point as done previously. The base station SDR can request any combination of the three text files, including just a single file, to be transmitted back towards the base station. The field sensor SDR will send each file completely, one-at-a-time, until the request is fulfilled. The repeat bug found earlier in GNU Radio Companion is not addressed at this stage of the project, and the consequences of its problems were addressed in this upgrade by having a short timer set for each file to prevent infinite repetition.

2.2.4 Round-Robin Communication

The further modifications made for the next iteration of communication between the two devices takes place entirely within the Python file. With the addition of two-way communication with multiple text files in the last upgrade, the changes here mostly relate to timing and addressing the repeat issue from GNU Radio Companion. Code was developed to address the repeat bug that takes advantage of the local text files on each host. The Python file will read from each file that is requested by the base station host, and record one line at a time from each file into a separate text file for transmission. The additional layer of reading from a file addresses the issue of repeating well, and it allows for this process to work (using the text file setup) without changing the base libraries of GNU Radio Companion.

Following the fix of the repeat bug, the code for a round-robin scheduling priority was implemented at this time in the research. Round-robin algorithms work by sending a single data point from multiple set options, and then going around each set sending the first single data point until you reach the second data point of the first set chosen. This process repeats, one-at-a-time for each data set (or text file, in this case) until all the data has been transmitted. This works in the Python code with timings set between the two SDRs to prevent transmission and reception on one device at the same time. Although this serves the purpose of establishing the first basic algorithm for this project, the next step in the code is to set up acknowledgements in the

communication, where after receiving any data or information, the receiving radio sends a receipt of data received back to the transmitting radio to ensure that no data was lost and needs to be sent again.

2.2.5 Weighted Round-Robin Communication

The next iteration of upgrade that was included was a series of smaller changes that resulted in a complete weighted round-robin algorithm, or a round-robin algorithm that takes in weight factors to give priority to the sensors that need it. As mentioned previously, the first of those changes was to implement acknowledgements. In this manner, after the field sensor radio sends a single data point to the base station radio, the base station responds with an ACK message to show that it has received the data properly. The field sensor radio waits until the ACK is acquired before sending the next piece of data in the list. This is important because it makes the communication channel more robust and decreases the likelihood of error in transmission. It also allows the algorithm to be less dependent on timing between the two radios, and instead allows them to be synced up like a conversation. At this point, much of the intentional delays added to ensure the timing of the radios were synced were removed and the communication process took much less time to complete.

Following this, the weights pertaining to each text file (and will later be each sensor) needed to be implemented. A user-input terminal request for weights was added, and after spending some time developing the code for a proper weighted round-robin algorithm for determining the order of files to send, the basics of weighted round-robin were present. The inclusion of the algorithm was successful and established the correct order for weighted round-robin, whether for two text files or for ten, by labeling each file in a list, calculating the proper

order based on the inputted weights, and ensuring that the next data point in the list to be sent had an identifier that matched the request from the algorithm.

After confirming the success of the ordering of data, the number of text files was extended to fourteen (still with fake data) to ensure readiness for the fourteen sensors with real data that are represented in this research. The real data was then included, replacing all the fake data that has been used up to this point, and additional code had to be written to address the formatting differences (all the real data was included in a single csv file). This was relatively easy to address compared with the other components of the algorithm.

For each sensor being represented, the next step to include for the weighted round-robin is a prediction scheme for guessing the next value that will be sent based on the data of the past. The prediction for the next value in each sensor was developed as the average of all of the data that has been sent previously, which was not the best prediction method but worked as a baseline. The significance of including this is that it allows a root-mean-square-error (RMSE) to be calculated and generated at the end of the communication between the radios for each sensor, and then added together to find a total RMSE. This is the metric that is used to evaluate the success of the models created. Final modifications are minor and include examples such as having the end of the file communicated to stop the communication, having the field sensor wait on the base station to be running before continuing operation, and others. The changes were successful, completing the requirements for the weighted round-robin algorithm that were desired.

2.2.6 Smart Weighted Round-Robin Communication

The last form up the algorithm developed is the smart weighted round-robin algorithm. The difference between the last version of this and the smart version is that the smart weighted

round-robin algorithm updates the weights in real time to get more data and make better predictions, resulting in a lower error rate. This final change required some restructuring of the previous Python codes to accommodate the changing weights, including additional communication back and forth between the radios among others.

On the base station radio side of the code, many updates were included. The first is the weight change calculations and parameters for making changes to the weight; this was accomplished with creative thinking and experimental testing of different techniques. The ending parameters established for weight changes are as follows: the weight of a respective sensor will increase by a factor of one if the difference between the prediction and real value is greater than two times the weight, and the weight will decrease by a factor of one if the difference is less than the weight. This allows a more dynamic change in weights than simply looking at the difference in prediction and real value; the weight factor implemented spreads priority more evenly over the sensors and prevents exponential growth or collapse in the weight changes. This method for changing the weights is quite stable, and by far gave the best results compared to other techniques. Lastly, the weights were transmitted back to the field sensor radio side of communication to implement the calculated changes.

The other major change to the base station code is the improved prediction model. As mentioned previously, the prediction for the next value at each sensor in the weighted roundrobin algorithm was built on the average of all data that had already transmitted. This was changed at this stage of the research project to a much better model: the average difference between two data points for every pair sent thus far is added to the last number received. The shifting of the average from the real value to the difference in real values was remarkably successful at preventing wide errors in prediction.

On the field sensor radio side, the only changes made were taking the weight updates that were sent every twenty transmissions and incorporating them into the new weighted round-robin order that the scheduler makes. This required some creativity, not only for the difficulty of implementing the reorder in further transmission, but for recognizing the data that has already been sent. This problem was solved by including fillers of zero in the first parts of the order that had already been transmitted, and from there, the smart weighted round-robin worked perfectly.

3. **RESULTS**

3.1 Operation of the Complete System

The physical setup of the architecture as described in Figure 1.4 is modeled below in Figure 3.1. Each side of the radios has a host device that the user interacts with, and the radios are communicating with the various algorithms described over-the-air.



Figure 3.1: Overview of Complete System

There were complications in achieving a UDP connection between the localhost device and the USRP radios. This aspect was deemed to be beyond the scope of this research project, so instead the smart weighted round-robin algorithm was developed for this project. The Python codes for the base station and field sensor radios are contained in the hard drive of their respective host devices, and the public repository data used for the sensor data is stored on the field sensor radio's host. Otherwise, the files and information necessary for communication are created during the operation of the Python software files themselves.

3.2 Effectiveness of Algorithms

For data collection, the process was relatively simple. The algorithms were each ran with 150 data points total, meaning that there were 150 transmissions from the field sensor radio to the base station radio and 150 sent acknowledgements in return. Because there are fourteen sensors, this is adequate amount of information to compare the effectiveness of the various algorithms, which is measured with the RMSE values from each sensor as well as the total RMSE for a single iteration. For the algorithms with weights, the weights are varied over several iterations to get a solid set of data for comparison. Although the round-robin original algorithm did not include a measurement for prediction and RMSE values, entering a value of one for the weight of every sensor in the weighted round-robin algorithm will give the values for a simple round-robin iteration. The various information collected is displayed in Table 3.1 below.

	Α	В	С	D	E	F	G	н	I	1	к	L	м	N	Total
Weight	1	1	1	1	1	1	1	1	1	1	1	1	1	1	-
WRR RMSE	37.958	48.737	25.271	9.933	24.029	23.744	24.61	17.505	5.069	9.488	12.225	9.097	5.805	4.411	257.883
Smart RMSE	8.769	6.5	15.955	2.557	12.362	4.981	9.725	30.095	6.935	12.817	27.549	6.728	3.615	2.242	150.83
Weight	1	6	4	8	3	2	6	1	4	6	3	5	5	1	-
WRR RMSE	7.92	88.451	25.271	16.044	19.298	12.833	30.203	13.796	5.069	13.341	11.136	10.563	8.784	0	262.708
Smart RMSE	8.393	6.498	16.211	4.274	12.12	4.981	10.54	26.98	6.935	13.248	27.549	6.111	3.623	0	147.462
Weight	3	2	5	2	1	3	3	5	2	4	1	2	3	4	-
WRR RMSE	42.058	29.871	75.265	7.861	8.229	23.744	24.61	22.83	4.316	12.268	7.437	6.379	6.939	6.142	277.95
Smart RMSE	8.769	6.498	16.365	2.478	10.537	4.983	10.214	34.058	6.819	13.248	21.375	5.453	3.473	5.526	149.794
Weight	2	4	2	1	7	1	4	5	3	6	1	6	2	3	-
WRR RMSE	21.353	63.283	12.653	3.249	36.015	5.843	25.565	19.743	4.499	15.805	7.437	13.2	3.133	4.219	235.996
Smart RMSE	8.748	6.401	15.955	2.26	13.259	4.677	10.214	34.058	6.935	13.287	22.631	6.823	3.206	1.099	149.553
Weight	6	3	3	5	2	2	1	3	4	1	2	4	1	2	-
WRR RMSE	90.045	56.005	29.443	14.159	19.298	18.668	12.515	17.67	6.739	6.346	11.02	11.486	2.174	3.006	298.573
Smart RMSE	8.77	6.498	16.211	4.077	12.12	4.981	9.167	31.966	6.935	12.065	27.133	5.453	3.206	1.099	149.681

Table 3.1: Final Data Results with Varying Weights

The format of the table has sensors A - N (all fourteen sensors) in the column labels at the top, and in groups of three rows, it shows the weights chosen for that specific iteration along with the root-mean-square-error values from the weighted round-robin (WRR RMSE) and the smart weighted round-robin (Smart RMSE). On the rightmost column, the total RMSE for each was also included.

3.2.1 Round-Robin Algorithm

The only relevant data from Table 3.1 for the basic round-robin algorithm is the first iteration, where all the weights are set to one, meaning the priority is spread across each sensor equally, which results in going around one-by-one in the round-robin manner. The total error that was calculated in this algorithm run is 257.883 from adding each sensor's error together; without a base reference, this feels meaningless. However, comparing to the other (randomly chosen) weight values, 257.883 is reasonably within the bounds of error from the other weighted round-robin total errors. It is not the highest or lowest error value, which makes sense because the varying weights are going to impact the errors for each sensor differently which can have a positive or negative impact on the value. This is true for the total values, but also for each of the sensors. For the remaining algorithms, the analysis can be done with the individual sensor error values and the total error from the basic round-robin as a base value.

3.2.2 Weighted Round-Robin Algorithm

Changing the weights with randomly chosen values resulted in four iterations of data as shown in Table 3.1 above (excluding the first iteration, representing basic round-robin). Keeping the values from the round-robin algorithm run as base values to compare to, some changes in individual sensor weights add to the RMSE value, while others decrease it. The variations all remain within reasonable bounds from each other, such as the fact that all the total RMSE values are in the two-hundreds. Because the data set for each iteration is the same, with some careful analysis, the results from the weighted round-robin algorithm can be used and further iterations can be ran to discover ideal values for individual weights leading towards the much lower error rates. Fortunately, this is unnecessary because the next algorithm does this on its own without any need for user calculations.

3.2.3 Smart Weighted Round-Robin Algorithm

The smart weighted round-robin algorithm makes drastic improvements on the RMSE values for every single iteration tested. When comparing results to the base value from basic round-robin in Table 3.1, the smart algorithm yields a total RMSE of 150.830, compared to the original RMSE in round-robin of 257.883, a 41.5% reduction of error. The trend holds out for the various weights attempted; the second through fifth iterations yield reductions of 43.9%, 46.1%, 36.6%, and 50.0% respectively. These improvements are massive and prove that the smart weighted round-robin algorithm consistently outperforms the basic round-robin algorithm and weighted round-robin algorithm drastically.

For most of the individual sensor data points, the trend also holds, although there is some variation where the weighted round-robin outperforms specific sensors. This is primarily due to two reasons: the first is that there is an inherent bias towards the first few sensors in the order, and the second is that the actual data is fluctuating in different manners. The bias exists because of the manner of scheduling in weighted round-robin; even if the weights are distributed equally, the first sensors still transmit their data first. When the updates to the weights are implemented and the new schedule is created for the remaining data, if any two weights are equal, the first sensor in the original order is prioritized. This bias has been mostly reduced with the update to the next-value prediction model, but it is still present at a small scale and likely will be for all variations of real-time weighted round-robin algorithm is that the real-time data from the public repository is fluctuating and inconsistent across all the sensors, so some predictions for the next-values will be more inaccurate than the mean of all data points when the data changes from increasing to decreasing or vice versa. This is inherently a property of the data used, and the

prediction would be more accurate over longer periods of time (meaning more data points in a transmission would reduce the chance of this occurring). Regardless, the total RMSE values are significantly lower than the other algorithms, so these edge cases are largely unimportant to the grand scheme of the communication.

4. CONCLUSION

4.1 Effectiveness of SDN-SDR Network

The combination of software-defined radios and software-defined networks into an experimental architecture such as done with this project is a task that is profound and fascinating in its development. Each technology serves to displace its predecessor by moving components into software, allowing greater flexibility with the construction of this network than previously. The radios themselves allow communication between the base station and the field sensors to take place over-the-air, and the elements of their communication controlled in software (frequency, sample size) can be changed in real-time. The software nature of the network itself allows code to be developed further for the virtual controller to make specific changes for real-world applications, such as a certain method of reading a set of sensors or changes to who can access the network.

Additionally, because everything other than the actual B210s takes place in software, once the radios are set up in permanent place as meant to be for the project, changes to the entire architecture can be made remotely. This is perhaps the most key benefit of the use of these newer technologies in the network construction – the ability to change settings remotely directly addresses the issues with traditional networking discussed in the introduction and increases the ease of use for the technology. For example, one can imagine a scenario where out of the thousands of sensors reading temperature, a critical sensor starts acting strange. With the SDR-SDN full setup, the sensor can be isolated, analyzed, fixed, and reincluded without a single shutdown of the entire network. The benefits of this are huge.

However, one critical aspect that should be discussed is the practicality of converting modern networks into the software domain using SDN and SDR technology. It is easy to imagine one or the other being used for its respective widening of the range of applications but implementing both at the same time is unlikely for a company or project to do, and there are potential benefits from that decision. The learning curve for creating and adapting this network, along with integrating the SDN and SDRs together on each host, is a huge cost to the construction of an SDR-SDN network that many projects will not be willing to undertake. The components of a traditional network, by comparison, are well-studied, well-understood, and widely applied so that there are infinite resources related to the construction of the network and network engineers that are trained to create them. With both software-defined networks and software-defined radios, each technology is documented decently well, but the user base is not large enough to have a comparable number of resources to use. When the technologies are combined, there are far fewer than there are independently.

Once the large costs for acquiring the equipment and understanding it to the point of development are undergone, however, the radios and network architecture as a whole are quite useful and applicable in the world. For projects that need specific flexibility and special attention, the SDN-SDR architecture can address those needs in a highly dynamic manner.

4.2 Real-World Applications

The real-world applications of SDNs, SDRs, and SDN-SDR systems are interesting to consider. As stated previously, the ability to control elements using software has been critical in the development of all sorts of modern technology – smart phones, cameras, laptops, and cars, to name a few examples. The changes to network and radio technologies are not unprecedented,

and the expectation is that the popularity of the other devices that use a combination of hardware and software elements will be repeated with these tools.

Software-defined networks, however, have had somewhat limited success in finding applications (Saleem 2016). For much of the world, the costs for implementing SDNs is much higher than the costs of operating a traditional network. Also, the surrounding network infrastructure has for a long time been underdeveloped to serve the purposes of SDNs, although the ground for this is changing as the demand for more network flexibility goes up. Despite the excitement for the technology, the rate of growth has not met expectations, and the SDN has not yet proven itself as a permanent upgrade to networking.

The software-defined radios have had more success in market implementation. The SDRs are applicable almost everywhere where radios were previously used, such as communication, measuring and monitoring data, and streaming information. The immediate bonus of using an SDR as opposed to a traditional radio is the ability to have multiple options for configuration for different scenarios. For testing equipment or analyzing data from sensors, the flexibility involved in being able to switch between predefined, custom settings allows much less radio technology to be used as many of their individual tasks have been taken instead by the SDR. For many industries and companies, particularly those involved with networks, SDRs are gaining popularity and have an optimistic future.

The real-world applications for SDN-SDR combinations are limited at the moment. For the industries that want to create networks with the highest flexibility, especially regarding sensor reading, the technology can be very useful. However, this has yet to occur beyond the research efforts behind the technology. One could imagine, however, that companies that are reading lots of sensor data in real-time would receive lots of benefits by investing in SDN-SDR

combinations for their data collection, as done in this research project. The modular nature of the project allows any number of sensors to be added, and each sensor can be remotely modified at any point in time. An industry such as weather prediction, with conditions that change rapidly and the requirement of a lot of sensor data, would benefit from using the SDN-SDR technology.

For now, the devices used in this research project are primarily at the research stage of their development, but the applications to real-world projects are promising.

REFERENCES

- [1] "The Basics of SDN and the OpenFlow Network Architecture." NoviFlow, NoviFlow, 6 May 2021, noviflow.com/the-basics-of-sdn-and-the-openflow-network-architecture/.
- [2] "What Is SDN?" *Juniper Networks*, Juniper Networks, https://www.juniper.net/us/en/research-topics/what-is-sdn.html.
- [3] Akeela, Rami, and Behnam Dezfouli. "Software-Defined Radios: Architecture, State-ofthe-Art, and Challenges." *Computer Communications*, Elsevier, 30 July 2018, https://www.sciencedirect.com/science/article/abs/pii/S0140366418302937.
- [4] Cooney, Michael. "What Is SDN and Where Software-Defined Networking Is Going." Network World, Network World, 16 April. 2019, www.networkworld.com/article/3209131/what-sdn-is-and-where-its-going.html.
- [5] Hamill, Stephanie. "Introduction to Software Defined Radio." Wireless Innovation Forum, 2021, www.wirelessinnovation.org/Introduction_to_SDR.
- [6] IBM Cloud Education. "The Fundamentals of Networking." *IBM*, IBM, 17 Mar. 2021, https://www.ibm.com/cloud/learn/networking-a-complete-guide#toc-important--ez_DoRa_.
- Jain, Raj. "OpenFlow, Software Defined Networking (SDN) and Network Function Virtualization." McKelvey School of Engineering. Tutorial at 2014 IEEE 15th International Conference on High Performance Switching and Routing, 2014, Vancouver, Canada.
- [8] Jena, Satyabrata. "Difference between Software Defined Network and Traditional Network." *GeeksforGeeks*, 21 Aug. 2020, https://www.geeksforgeeks.org/difference-between-software-defined-network-and-traditional-network/.
- [9] Lessing, Marlese, and Connor Craven. "What Is an SDN Controller? Definition." SDxCentral, SDxCentral Studios, 2020, www.sdxcentral.com/networking/sdn/definitions/what-is-sdn-controller/.
- [10] Molenaar, Rene. "Introduction to SDN (Software Defined Networking)."

NetworkLessons.com, 19 Dec. 2019, networklessons.com/cisco/ccna-routing-switching-icnd2-200-105/introduction-to-sdn-software-defined-networking.

[11] Saleem, Maab. "Software Defined Network: Use Cases From the Real World" RouterFreak.com, 14 July. 2016, https://www.routerfreak.com/software-defined-networkuse-cases-from-the-real-world/.

APPENDIX A: PYTHON SOFTWARE WRITTEN FOR

COMMUNICATION

1	#:/ugr/bin/env python2	
2	# -*- ooding: utf-8 -*-	
3	######################################	
5	* GNV RALLO FYLDON FLOW GRADN # Title: Chat	
6	# Generated: Thu Feb 10 09:30:28 2022	
7		
8		
9	ifname == 'main':	
10 11	import ctypes	
12	<pre>import sys if sys.platform.startswith('linux'):</pre>	
13	try:	
14	<pre>x11 = ctypes.cdll.LoadLibrary('libX11.so')</pre>	
15	x11.XInitThreads()	
16	except:	
17 18	print "Warning: failed to XInitThreads()"	
19	import	
37		
38		
39	class chat(gr.top_block, Qt.QWidget):	
40 41	definit(self):	
42	<pre>driv(self, "Chat") gr.top blockinit(self, "Chat")</pre>	
43	Ot.OWidgetinitself)	
44	self.setWindowTitle("Chat")	
45	<pre>qtgui.util.check_set_qss()</pre>	
46 47	try:	
48	<pre>self.setWindowIcon(Qt.QIcon.fromTheme('gnuradio-gro')) except:</pre>	
49	pass	
50	<pre>self.top_scroll_layout = Qt.QVBoxLayout()</pre>	
51	self.setLayout(self.top_scroll_layout)	
52	self.top_scroll = Qt.QScrollArea()	
53 54	self.top_scroll.setFrameStyle(Qt.QFrame.NoFrame) self.top_scroll_layout.addWidget(self.top_scroll)	
55	self.top_scroll.setWidgetResizable(True)	
56	<pre>self.top_widget = Qt.QWidget()</pre>	
57	self.top_scroll.setWidget(self.top_widget)	
58	<pre>self.top_layout = Qt.QVBoxLayout(self.top_widget) <pre>self.top_arcd_layout(self.top_widget)</pre></pre>	
59 60	<pre>self.top_grid_layout = Qt.QGridLayout() self.top_layout.addLayout(self.top_grid_layout)</pre>	_
61		
62	<pre>self.settings = Qt.QSettings("GNU Radio", "chat")</pre>	
63	<pre>self.restoreGeometry(self.settings.value("geometry").toByteArray())</pre>	
64		
65 66		
67	# Variables	
68	2	
69	<pre>self.samp_rate = samp_rate = 400000</pre>	
70		
71 72	<pre>while True: weight1 = raw_input("Enter a value for the weight of sensor 1 (integer): ")</pre>	
73	try:	
74	int(weight1)	
75	<pre>if (int(weight) < 0):</pre>	
76	print "Value cannot be negative. Try again."	
77 78	continue except ValueError:	
79	print "Value provided needs to be an integer. Try again."	
80	continue	
81	break	
82		
83 84	while True:	
85	<pre>while irue: weight2 = raw input("Enter a value for the weight of sensor 2 (integer): ")</pre>	
86	try:	
87	int (weight2)	
88	if (int(weight2) < 0):	
89 90	print "Value cannot be negative. Try again." continue	
90	except ValueError:	
	· · · · · · · · · · · · · · · · · · ·	

92	print "Value provided needs to be an integer. Try again."	
93	continue	
94	break	
95		
96 97	<pre>while True: weight3 = raw_input("Enter a value for the weight of sensor 3 (integer): ")</pre>	
98	try:	
99	int(weight3)	
100	<pre>if (int(weight3) < 0);</pre>	
101	print "Value cannot be negative. Try again."	
102	ontinue	
103 104	except ValueError: print "Value provided needs to be an integer. Try again."	
105	continue	
106	break	
107		
108	while True:	
109 110	<pre>weight4 = raw_input("Enter a value for the weight of sensor 4 (integer): ") U try:</pre>	
111	int(weight4)	
112	if (int(weight4) < 0):	
113	print "Value cannot be negative. Try again."	
114	Continue	
115 116	<pre>except ValueError: print "Value provided needs to be an integer. Try again."</pre>	
117	continue	
118	break	
119		
120	while True:	
121 122	<pre>weight5 = raw_input("Enter a value for the weight of sensor 5 (integer): ") torm:</pre>	
122	try: int(weight5)	
124	if (int(weight5) < 0):	
125	print "Value cannot be negative. Try again."	
126	continue	
127 128	except ValueError: print "Value provided needs to be an integer. Try again."	
120	continue	
130	break	
131		
132	while True:	
133	<pre>weight6 = raw_input("Enter a value for the weight of sensor 6 (integer): ") temp</pre>	
134 135	<pre>try: int(weight6)</pre>	
136	<pre>if (int/weight6) < 0):</pre>	
137	print "Value cannot be negative. Try again."	
138	e continue	
139	except ValueError:	
140 141	print "Value provided needs to be an integer. Try again."	
142	brak	
143		
144	• while True:	
145 146	<pre>weight7 = raw_input("Enter a value for the weight of sensor 7 (integer): ") try:</pre>	
140	int (weight7)	
148	if (int(weight7) < 0):	
149	print "Value cannot be negative. Try again."	
150	continue continue	
151 152	except ValueError:	
152	print "Value provided needs to be an integer. Try again."	
154	brak	
155		
156	while True:	
157 158	<pre>weight8 = raw_input("Enter a value for the weight of sensor 8 (integer): ") trutter to be a sensor a sens</pre>	
158	try: int(weight8)	
160	<pre>in((weight8) < 0):</pre>	
161	print "Value cannot be negative. Try again."	
162	print "Value cannot be negative. Try again." continue	
162 163	print "Value cannot be negative. Try again." continue except ValueError:	
162	print "Value cannot be negative. Try again." continue	

166	break	
167		
168	while True:	
169	weight9 = raw_input("Enter a value for the weight of sensor 9 (integer): ")	
170 (try:	
171	int(weight9)	
172	if (int(weight9) < 0):	
173	print "Value cannot be negative. Try again."	
174	continue	
175	except ValueError:	
176	print "Value provided needs to be an integer. Try again."	
177	continue	
178 (break	
179		
180 (181	<pre>while True: weight10 = raw_input("Enter a value for the weight of sensor 10 (integer): ")</pre>	
182		
183	try: int(weight10)	
184	if (int(weight10) < 0):	
185	print "Value cannot be negative. Try again."	
186 (continue	
187	except ValueError:	
188	print "Value provided needs to be an integer. Try again."	
189	continue	
190	break	
191		
192	while True:	
193	weight11 = raw input("Enter a value for the weight of sensor 11 (integer): ")	
194	try:	
195	int (weight11)	
196	<pre>if (int(weight11) < 0):</pre>	
197	print "Value cannot be negative. Try again."	
198 (continue	
199 (except ValueError:	
200	print "Value provided needs to be an integer. Try again."	
201 (continue	
202 [break	
203		
204 (while True:	
205	weight12 = raw_input("Enter a value for the weight of sensor 12 (integer): ")	
206 (try:	
207	int(weight12)	
208 [
209	print "Value cannot be negative. Try again."	
	continue	
211 (except ValueError:	
212	print "Value provided needs to be an integer. Try again."	
213	continue continue	
214 (break	
215		
216	while True:	
217	<pre>weight13 = raw_input("Enter a value for the weight of sensor 13 (integer): ")</pre>	
218 (try:	
219	int(weight13)	
220 (<pre>if (int(weight13) < 0):</pre>	
221	print "Value cannot be negative. Try again."	
222	continue	
223	except ValueError:	
224	print "Value provided needs to be an integer. Try again."	
225	continue break	
226 (227	JE SAN	
227 228 (while True:	
228 (<pre>while inue: weight14 = raw input("Enter a value for the weight of sensor 14 (integer): ")</pre>	
229 230 (weight14 = raw_input("Enter a value for the weight of sensor 14 (integer): ") try:	
230 (try: int(weight14)	
231 232 (<pre>int(Weightl4) < 0):</pre>	
232 (print "Value cannot be negative. Try again."	
235 234 (continue	
234 (excent ValueFror:	
236	print "Value provided needs to be an integer. Try again."	
236 237 (continue	
238 (- brak	
239		

240			
		<pre>weight_file = open('/home/c/chat3', 'w')</pre>	
241			
242		weight_file.write(weight1 + ',')	
243		weight file.write(weight2 + ',')	
244		weight file.write(weight3 + ',')	
245		weight_file.write(weight4 + ',')	
246		weight file.write(weight5 + ',')	
247		weight file.write(weight6 + ',')	
248		<pre>weight_file.write(weight7 + ',') </pre>	
249		weight_file.write(weight8 + ',')	
250		weight_file.write(weight9 + ',')	
251		weight_file.write(weight10 + ',')	
252		weight_file.write(weight11 + ',')	
253		weight_file.write(weight12 + ',')	
254		weight file.write(weight13 + ',')	
255		weight file.write(weight14 + ',\n')	
256			
257		weight_file.close()	
258			
259 (
260		# Elooks	
261 (
262		self.uhd_usrp_sink_1 = uhd.usrp_sink(
263		",".join(("", "")),	
264		uhd.stream_args(
265		cpu_format="fo32",	
266		channels=range(1),	
267),	
268)	
269		self.uhd usrp sink 1.set samp rate(samp rate)	
270		self.uhd_usrp_sink_1.set_time_now(uhd.time_spec(time.time()), uhd.ALL_MEOARDS)	
271		self.uhd_usrp_sink_1.set_center_freq(1000000000, 0)	
272		self.uhd usrp sink 1.set gain(40, 0)	
273		self.uhd usrp sink 1.set antenna('RX2', 0)	
274		<pre>self.digital_gmsk_mod_0 = digital.gmsk_mod(</pre>	
275		<pre>samples_per_symbol=2, bt=0.35,</pre>	
276			
277		verbose=False,	11
278		log=False,	
279)	
280		self.blocks_throttle_0 = blocks.throttle(gr.sizeof_char*1, samp_rate,True)	
281		<pre>self.blocks_multiply_const_vxx_1 = blocks.multiply_const_vcc((1,))</pre>	
282		<pre>self.blocks_head_0 = blocks.head(gr.sizeof_char*1, 100000)</pre>	
283		<pre>self.blocks_file_source_0 = blocks.file_source(gr.sizeof_char*1, '/home/c/chat3', True)</pre>	
284		self.blocks_file_source_0.set_begin_tag(pmt.PMT_NIL)	
285		self.blks2_packet_encoder_0 = grc_blks2.packet_mod_b(grc_blks2.packet_encoder(
286			
287		samples per symbol=2,	
		samples_per_symbol=2, bits per_symbol=1.	
		bits_per_symbol=1,	
		<pre>bits_per_symbol=1, preamble='',</pre>	
289		<pre>bits_per_symbol=1, preamble='', access_code='',</pre>	
289 290		<pre>bits_per_symbol=1, preamble='', access_code='', pad_for_usrp=True,</pre>	1
289 290 291		<pre>bits_per_symbol=1, preamble='', access_code='', pad_for_usrp=True,),</pre>	1
289 290 291 292		<pre>bits_per_symbol=1, preamble=`', access_code=`', pad_for_usrp=True,), payload_length=0,</pre>	1
289 290 291 292 293		<pre>bits_per_symbol=1, preamble='', access_code='', pad_for_usrp=True,),</pre>	1
289 290 291 292 293 294		<pre>bits_per_symbol=1, preamble=`', access_code=`', pad_for_usrp=True,), payload_length=0,</pre>	I
289 290 291 292 293 294 295		<pre>bits_per_symbol=1, preamble=`', access_code=`', pad_for_usrp=True,), payload_length=0,</pre>	1
289 290 291 292 293 294 295 296		<pre>bits_per_symbol=1, preamble='', access_code='', pad_for_usrp=True,), payload_length=0,)</pre>	1
289 290 291 292 293 294 295 296		<pre>bits_per_symbol=1, preamble=`', access_code=`', pad_for_usrp=True,), payload_length=0,</pre>	I
289 290 291 292 293 294 295 295 296 297		<pre>bits_per_symbol=1, preamble='', access_code='', pad_for_usrp=True,), payload_length=0,)</pre>	1
289 290 291 292 293 294 295 295 296 297		<pre>bits_per_symbol=1, preamble='', access_code='', pad_for_usrp=True,), payload_length=0,)</pre>	1
289 290 291 292 293 294 295 296 297 298 299		<pre>bits_per_symbol=1, preamble='', access_code='', pad_for_usrp=True,), payload_length=0, } ; ; connections</pre>	I
289 290 291 292 293 294 295 296 297 298 299		<pre>bits_per_symbol=1, preamble='', access_code='', pad_for_usrp=True,), payload_length=0,) # Connections # Connections</pre>	1
289 290 291 292 293 294 295 296 297 298 299 300		<pre>bits_per_symbol=1, preamble='', access_code='', pad_for_usrp=True,), payload_length=0,) / /////////////////////////////</pre>	I
289 290 291 292 293 294 295 296 297 298 299 300 301		<pre>bits_per_symbol=1, preamble='', access_code='', pad_for_usrp=True,), payload_length=0,)</pre>	I
289 290 291 292 293 294 295 296 297 298 299 300 301 302		<pre>bits_per_symbol=1, preamble='', access_code='', pad_for_usrp=True,), payload_length=0,)</pre>	I
289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304		<pre>bits_per_symbol=1, preamble='', access_code='', pad_for_usrp=True,), payload_length=0,)</pre>	
289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 303 304 305		<pre>bits_per_symbol=1, preamble='', access_code='', pad_for_usrp=True,), payload_length=0,)</pre>	1
289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 304 305 306	Q 6 7	<pre>bits_per_symbol=1, preamble='', access_code='', pad_for_usrp=True,), payload_length=0,)</pre>	1
289 290 291 292 293 294 295 296 296 299 300 301 302 303 304 305 305 306 307	⊽ ≙ ⊖ ⊽ de:	<pre>bits_per_symbol=1, preamble='', access_code='', pad_for_usrp=True,), payload_length=0,)</pre>	
289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308	 □ □ □ de: 	<pre>bits_per_symbol=1, preamble='', access_cod='', pad_for_usrp=True,), payload_length=0,)) # Connections # Connect((self.blks2_packet_encoder_0, 0), (self.digital_gmsk_mod_0, 0)) self.connect((self.blcks_ite_source_0, 0), (self.blcks_throttle_0, 0)) self.connect((self.blccks_multiply_const_vxx_1, 0), (self.uhd_usrp_sink_1, 0)) self.connect((self.blccks_multiply_const_vxx_1, 0), (self.blcks_multiply_const_vxx_1, 0)) self.connect((self.blccks_multiply_const_vxx_1, 0), (self.blcks_multiply_const_vxx_1, 0)) self.connect((self.digital_gmsk_mod_0, 0), (self.blcks_multiply_const_vxx_1, 0)) self.connect((self.digital_gmsk_mod_0, 0), (self.blcks_multiply_const_vxx_1, 0)) closeEvent(self, event): self.settings = Qt.QSettings("GNU Radio", "chat")</pre>	
289 290 291 292 293 294 295 296 297 298 299 300 300 300 300 300 300 300 300 300 3	↓	<pre>bits_per_symbol=1, preamble='', access_code='', pad_for_usrp=True,), payload_length=0,)</pre>	
302 303 304 305 306 307 308 309 310	マ □ □ □ □ □	<pre>bits_per_symbol=1, preamble='', access_cod='', pad_for_usrp=True,), payload_length=0,)) # Connections # Connect((self.blks2_packet_encoder_0, 0), (self.digital_gmsk_mod_0, 0)) self.connect((self.blcks_ite_source_0, 0), (self.blcks_throttle_0, 0)) self.connect((self.blccks_multiply_const_vxx_1, 0), (self.uhd_usrp_sink_1, 0)) self.connect((self.blccks_multiply_const_vxx_1, 0), (self.blcks_multiply_const_vxx_1, 0)) self.connect((self.blccks_multiply_const_vxx_1, 0), (self.blcks_multiply_const_vxx_1, 0)) self.connect((self.digital_gmsk_mod_0, 0), (self.blcks_multiply_const_vxx_1, 0)) self.connect((self.digital_gmsk_mod_0, 0), (self.blcks_multiply_const_vxx_1, 0)) closeEvent(self, event): self.settings = Qt.QSettings("GNU Radio", "chat")</pre>	
289 290 291 292 293 294 295 296 299 300 301 302 303 304 305 306 306 307 308 309 309 310		<pre>bits_per_symbol=1, preamble='', access_cod='', pad_for_usrp=True,), payload_length=0,)) # Connections # Self.connect((self.blxs2_packet_encoder_0, 0), (self.digital_gmsk_mod_0, 0)) self.connect((self.blocks_inel_source_0, 0), (self.blocks_throttle_0, 0)) self.connect((self.blocks_multiply_const_vxx_1, 0), (self.uhd_usrp_sink_1, 0)) self.connect((self.blocks_multiply_const_vxx_1, 0), (self.blocks_multiply_const_vxx_1, 0)) self.connect((self.blocks_multiply_const_vxx_1, 0), (self.blocks_multiply_const_vxx_1, 0)) self.connect((self.digital_gmsk_mod_0, 0), (self.blocks_multiply_const_vxx_1, 0)) self.connect((self.event): self.settings = Qt.QSettings("GNU Radio", "chat") self.settings.setValue("geometry", self.saveGeometry()) event.accept()</pre>	
289 290 291 292 293 294 295 296 297 298 299 300 300 300 300 300 303 304 305 306 306 307 308 309 310		<pre>bits_per_symbol=1, preamble='', access_code='', pad_for_usrp=True,), payload_length=0,)</pre>	

	314		
<pre>11 </pre>			<pre>def set_samp_rate(self, samp_rate):</pre>
	316		
<pre></pre>	317		
<pre>sets tex_rn(y:.tp_lbck, v:.dlige;): defnic(ni:Tx_Tx_Tv_) defnic(ni:_Tx_Tx_Tv_) defnic(ni:_Tx_Tx_Tv_) defnic(ni:_Tx_Tx_Tv_) defnic(ni:_Tx_Tx_Tv_) defnic(ni:_Tx_Tv_) defnic(ni:_Tx_Tv_)</pre>			<pre>self.blocks_throttle_0.set_sample_rate(self.samp_rate)</pre>
image description to place the place of the pla			
<pre>prove the set of the set of</pre>			
<pre>sis 0</pre>		cla	s text_rx(gr.top_block, Qt.QWidget):
int int <td></td> <td></td> <td>dat init (salt).</td>			dat init (salt).
0.0001200000000000000000000000000000000			
<pre>set_set_set_set_set_set_set_set_set_set_</pre>			
<pre>ist vertications(0, dice.formtheme('gymendic-grou') ist verticati</pre>	326		
<pre>ps</pre>	327		<pre>qtgui.util.check_set_qss()</pre>
<pre>second: function for the function of the</pre>	328		try:
<pre>sh</pre>			
<pre>sel _ sel _ resp _ serol l jayout = Qt.QtMonkayout () selstyperol = Qt.QtMonkayout () sel</pre>			
<pre>s1 = elst_ayout_elit.og_eccl_layout) s1 = elit.og_eccl_layout, 000 (elit.og_eccl_layout) s2 = elit.og_eccl_layout, 000 (elit.og_eccl_layout) s3 = elit.og_eccl_layout, 000 (elit.og_eccl_layout) s3 = elit.og_eccl_layout, 000 (elit.og_eccl_layout) s4 = elit.og_eccl_layout, 000 (elit.og_eccl_layout) s5 = elit.og_eccl_layout, 000 (elit.og_eccl_layout) s6 = elit.od_eccl_layout, 0000 (elit.og_eccl_layout) s6 = elit.od_eccl_layout, 0000 (elit.og_eccl_layout) s6 = elit.od_eccl_layout</pre>			
<pre>set .exp_=real = 0:.0⁺coltare() if .exp_=real.=0:.0⁺coltare() set .exp=.exp_=real.=0:.0⁺coltare() set .exp=.exp_=real.=0:.0⁺coltare() set .exp=.exp_=real.=0:.0⁺coltare() set .exp=.exp_=real.=0:.0⁺coltare() set .exp=.exp_=real.=0:.0⁺coltare() set .exp=.exp_=real.=0:.0⁺coltare(</pre>			
<pre>shi = sli.tep_scoletFrameNyte(0: OFFCame) slites() slites()</pre>			
<pre>set = set.fog_ercol_layout.addidge(=1.tog_ercol) set.fog_ercol_layout.addidge(=1.tog_ercol) set.fog_ercol_set.fog_ercol_set.fog_ercols set.fog_ercol_set.fog_ercols set.fog_ercol_set.fog_ercols set.fog_ercolset.fog_ercolset.fog_ercolset.fog_ercolset set.fog_ercolset.fog_erc</pre>			
<pre>sel.com_conl_setMidpetEstable(True) is est.com_dudget(stable(true) is est.com_dudget(sta</pre>	336		
<pre>self.top_accoll.setHidget(self.top_widget) self.top_accoll.setHidget(self.top_widget) self.top_accoll.setHidget(self.top_widget) self.top_accoll.setHidget(self.top_widget) self.top_accoll.setHidget(self.top_widget) self.top_accoll.setHidget(self.top_widget) self.top_accoll.setHidget(self.top_self.top) self.top_accoll.setHidget(self.top) self.top_a</pre>	337		
<pre>self.top_Tayout = 0.:0TextAyout(=01:top_widget)) self.top_Tayout = 0.:0TextAyout(=01:top_widget)) self.top_Tayout.addayout(=01:top_grid_layout) self.top_Tayout.addayout(=01:top_grid_layout)) self.top_Tayout.addayout(=01:top_grid_layout) self.top_Tayout.addayout(=01:top_grid_layout)) self.top_Tayout.addayout(=01:top_grid_layout)) self.top_Tayout.addayout(=01:top_grid_layout)) self.top_Tayout.addayout(=01:top_grid_layout)) self.top_Tayout.addayout(=01:top_grid_layout)) self.top_Tayout.addayout(=01:top_grid_layout)) self.top_Tayout.addayout(=01:top_grid_layout)) self.top_Tayout.adda</pre>			
<pre>shi too_rig_rig_layout of the set of occidatory of the set of</pre>			
<pre>si2 self.tsp_layout.addLayout(self.tsp_prid_layout) self.tsp_layout.addLayout(self.tsp_rid_layout) self.tsp_layout.addLayout(self.tsp_rid_layout) self.tspling = 00.0ettings("ON Madio", "tsgling").tsplithray()) self.tsgling = 00.0ettings("ON Madio", "tsgling").tsplithray() self.tsgling = 00.0ettings("ON Madio", "tsgling") self.tsgling</pre>			
<pre>30</pre>			
<pre>set solt.ubd_urp_source_0.set_map_tate(sam_tate) set.ubd_urp_source_0.set_mamp_tate(sam_tate) set.ubd_urp_source_tate(sam_tate) set.ubd_urp_source_tate(sam_tate) set.ubd_urp_source_tate(sam_tate) set.ubd_urp_source_tate(sam_tate) set.ubd_urp_source_tate(sam_tate) set.ubd_urp_source_tat</pre>			self.cop_rayout.addaayout(self.cop_grud_rayout)
<pre>shi self.restoredecometry(self.settings.value('geometry').toByteArray()) shi self.restoredecometry('geometry').toByteArray() shi self.restoredecometry('geometry').toByteArray()) shi self.restoredecometry('geometry').toByteArray() shi self.restoredecometry('geometry').toByteArray() shi self.restoredecometry('geometry').toByteArra</pre>			self.settings = Ot.OSettings("GNU Radio", "text rx")
<pre>start s</pre>			
<pre>sta</pre>	346		
199 if Availables 100 fift###################################	347		
<pre>soo c</pre>			
<pre>self.gamp_rate = gamp_rate = 400000 f ###############################</pre>			
<pre>32</pre>			
<pre>sis</pre>			self.samp_rate = samp_rate = 400000
<pre>press</pre>			
<pre>959 0 fffffffffffffffffffffffffffffffffff</pre>			
<pre>"".join((", ""), interface interface gen_form ifform ifform channels=range(1), if channels=range(1), if c</pre>			
<pre>bis bis bis bis bis bis bis bis bis bis</pre>	356		self.uhd_usrp_source_0 = uhd.usrp_source(
<pre>359 cpu_format="fo32", channels=range(1), 361), 362) 363 self.uhd_usrp_source_0.set_samp_rate(samp_rate) 364 self.uhd_usrp_source_0.set_gain(30, 0) 365 self.uhd_usrp_source_0.set_gain(30, 0) 366 self.uhd_usrp_source_0.set_gain(30, 0) 367 self.gtgui_freq_sink_x_0 = qtgui.freq_sink_c(368 1024, fsice 369 firdes.WIN_BLACKNAN_hARRIS, fwintype 370 0, ffo 371 samp_rate, fbw 373 1 fnumber of inputs 373 1 fnumber of inputs 374) 375 self.qtgui_freq_sink_x_0.set_update_time(0.10) 376 self.qtgui_freq_sink_x_0.set_update_time(0.10) 377 self.qtgui_freq_sink_x_0.set_y_alke(!Relative Gain', 'db') 378 self.qtgui_freq_sink_x_0.set_freque:(RELATED SERE, 0.0, 0, "") 379 self.qtgui_freq_sink_x_0.set_freque:(RELATED SERE, 0.0, 0, "") 381 self.qtgui_freq_sink_x_0.set_freque:(RELATED SERE, 0.0, 0, "") 382 self.qtgui_freq_sink_x_0.set_freque:(RELATED SERE, 0.0, 0, "") 383 self.qtgui_freq_sink_x_0.set_freque:(RELATED SERE, 0.0, 0, "") 384 self.qtgui_freq_sink_x_0.set_freque:(RELATED SERE, 0.0, 0, "") 385 self.qtgui_freq_sink_x_0.set_freque:(RELATED SERE, 0.0, 0, "") 383 self.qtgui_freq_sink_x_0.set_freque:(RELATED SERE) 384 self.qtgui_freq_sink_x_0.set_freque:(RELATED SERE) 383 self.qtgui_freq_sink_x_0.set_freque:(RELATED SERE) 384 self.qtgui_freq_sink_x_0.set_freque:(RELATED SERE) 385 self.qtgui_freq_sink_x_0.set_freque:(RELATED SERE) 385 self.qtgui_freq_sink_x_0.set_freque:(RELATED SERE) 384 self.qtgui_freq_sink_x_0.set_freque:(RELATED SERE) 385 self.qtgui_freq_sink_x_0.set_freqUE SERE SERE SERE SERE SERE SERE SERE SE</pre>	357		",".join(("", "")),
<pre>30</pre>	358		uhd.stream_args(
<pre>361), 362) 363 self.uhd_usrp_source_0.set_samp_rate(samp_rate) 364 self.uhd_usrp_source_0.set_gain(30, 0) 365 self.uhd_usrp_source_0.set_gain(30, 0) 366 self.uhd_usrp_source_0.set_gaintenne("X2", 0) 367 self.gtgui_freq_sink_x_0 = qtgui.freq_sink_c(368 1024, fsize 369 (1024, fsize 369 (1024, fsize) 370 0, ffo 371 samp_rate, fbw 372 ", fname 373 1 fnumber of inputs 374) 375 self.gtgui_freq_sink_x_0.set_update_time(0.10) 376 self.gtgui_freq_sink_x_0.set_y_label("Nelative Gain', 'db') 377 self.gtgui_freq_sink_x_0.set_y_label("Nelative Gain', 'db') 378 self.gtgui_freq_sink_x_0.set_y_label("Nelative Gain', 'db') 379 self.gtgui_freq_sink_x_0.set_gridoe(gtgui.TRIG_MODE_FREF, 0.0, 0, "") 381 self.gtgui_freq_sink_x_0.set_ft_average(1.0) 382 self.gtgui_freq_sink_x_0.self.gtaie(1.0) 383 self.gtgui_freq_sink_x_0.self.gtaie(1.0) 384 self.gtgui_freq_sink_x_0.self.gtaie(1.0) 385 self.gtgui_freq_sink_x_0.self.gtaie(1.0) 383 self.gtgui_freq_sink_x_0.sele]control_panel(False) 384 self.gtgui_freq_sink_x_0.senable_actorele(False) 385 s</pre>			
362) 363 self.uhd_usrp_source_0.set_samp_rate(samp_rate) 364 self.uhd_usrp_source_0.set_gain(30, 0) 365 self.uhd_usrp_source_0.set_antenna('RX2', 0) 366 self.uhd_usrp_source_0.set_antenna('RX2', 0) 367 self.qtgui_freq_sink_x_0 = qtgui.freq_sink_c(368 firdes.WIN_BLACKNN_hARRIS, fwintype 369 0, ffo 370 0, ffo 371 samp_rate, fbw 372 "', fname 373 1 fnumber of inputs 374) 375 self.qtgui_freq_sink_x_0.set_update_time(0.10) 376 self.qtgui_freq_sink_x_0.set_update_time(0.10) 377 self.qtgui_freq_sink_x_0.set_update_time(0.10) 378 self.qtgui_freq_sink_x_0.set_update_time(0.10) 379 self.qtgui_freq_sink_x_0.set_update_time(0.10) 371 self.qtgui_freq_sink_x_0.set_update_time(0.10) 372 self.qtgui_freq_sink_x_0.set_update_time(0.10) 373 self.qtgui_freq_sink_x_0.set_update_time(0.10) 374) 375 self.qtgui_freq_sink_x_0.set_update_time(0.10) 376 self.qtgui_freq_sink_x_0.set_u			
<pre>333</pre>),
364 self.uhd_usrp_source_0.set_center_freq(100000000, 0) 365 self.uhd_usrp_source_0.set_gain(30, 0) 366 self.uhd_usrp_source_0.set_gain(30, 0) 366 self.qtgui_freq_sink_x_0 = qtgui.freq_sink_c(368 self.qtgui_freq_sink_x_0 = qtgui.freq_sink_c(369 firdes.WIN_BLACKMAN_hARRIS, #vintype 370 0, ffo 371 samp_rate, #bv 372 "", fname 373 1 #number of inputs 374 > 375 self.qtgui_freq_sink_x_0.set_update_time(0.10) 376 self.qtgui_freq_sink_x_0.set_y_akie(140, 10) 377 self.qtgui_freq_sink_x_0.set_y_akie(140, 10) 378 self.qtgui_freq_sink_x_0.set_y_label(Relative Gain', 'dB') 379 self.qtgui_freq_sink_x_0.set_priger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "") 378 self.qtgui_freq_sink_x_0.enable_autoscale(False) 380 self.qtgui_freq_sink_x_0.enable_gail(Palse) 381 self.qtgui_freq_sink_x_0.enable_axis_labels(True) 382 self.qtgui_freq_sink_x_0.enable_control_panel(Palse) 383 self.qtgui_freq_sink_x_0.enable_control_panel(Palse) 384 self.qtgui_freq_sink_x_0.enable_control_panel(Pal) self ubd ugen source () set samn rate(samn rate)
365self.uhd_usrp_source_0.set_gain(30, 0)366self.uhd_usrp_source_0.set_antenna('KX2', 0)367self.uhd_usrp_source_0.set_antenna('KX2', 0)3681024, fsice3691024, fsice369firdes.WIN_BLACKMAN_HARRIS, fwintype3700, ffo371samp_rate, fbw372"", fname3731 fnumber of inputs374>375self.qtgui_freq_sink_x_0.set_update_time(0.10)376self.qtgui_freq_sink_x_0.set_vaxis(-140, 10)377self.qtgui_freq_sink_x_0.set_vaxis(-140, 10)378self.qtgui_freq_sink_x_0.set_vaxis(-140, 10)379self.qtgui_freq_sink_x_0.set_vaxis(-140, 10)370self.qtgui_freq_sink_x_0.set_vaxis(-140, 10)371self.qtgui_freq_sink_x_0.set_vaxis(-140, 10)372self.qtgui_freq_sink_x_0.set_vaxis(-140, 10)373self.qtgui_freq_sink_x_0.set_vaxis(-140, 10)374self.qtgui_freq_sink_x_0.set_trigger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "")375self.qtgui_freq_sink_x_0.set_trigger_mode(fstalse)380self.qtgui_freq_sink_x_0.set_trigger_mode(fstalse)381self.qtgui_freq_sink_x_0.set_fraverage(1.0)382self.qtgui_freq_sink_x_0.set_fraverage(1.0)383self.qtgui_freq_sink_x_0.enable_control_panel(False)384self.qtgui_freq_sink_x_0.enable_control_panel(False)384self.qtgui_freq_sink_x_0.enable_control_panel(False)385if not True:			
366 self.uhd_usr_source_0.set_antenna('RX2', 0) 367 self.qtgui_freq_sink_x_0 = qtgui.freq_sink_c(368 1024, fsize 369 firdes.WIN_BLACKMAN_hARRIS, #vintype 370 0, fso 371 samp_rate, fbv 372 "", fname 373 1 fnumber of inputs 374) 375 self.qtgui_freq_sink_x_0.set_update_time(0.10) 376 self.qtgui_freq_sink_x_0.set_update_time(0.10) 377 self.qtgui_freq_sink_x_0.set_update_time(0.10) 376 self.qtgui_freq_sink_x_0.set_update_time(0.10) 377 self.qtgui_freq_sink_x_0.set_leave 378 self.qtgui_freq_sink_x_0.set_leave 379 self.qtgui_freq_sink_x_0.set_leave 381 self.qtgui_freq_sink_x_0.enable_avtocale(False) 382 self.qtgui_freq_sink_x_0.enable_grid(False) 383 self.qtgui_freq_sink_x_0.enable_control_panel(False) 384 self.qtgui_freq_sink_x_0.enable_control_panel(False) 385 if not True:			
367 self.gtgui_freq_sink_x0 = qtgui.freq_sink_c(368 1024, fsize 369 firdes.WIN_BLACKMAN_hARRIS, fwintype 369 0, ffo 371 samp_rate, fbw 372 "", fname 373 1 fnumber of inputs 374) 375 self.qtgui_freq_sink_x0.set_update_time(0.10) 376 self.qtgui_freq_sink_x0.set_y_axis(-140, 10) 377 self.qtgui_freq_sink_x0.set_y_axis(-140, 10) 378 self.qtgui_freq_sink_x0.set_trigger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "") 379 self.qtgui_freq_sink_x0.set_figger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "") 381 self.qtgui_freq_sink_x0.set_figger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "") 382 self.qtgui_freq_sink_x0.set_figger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "") 383 self.qtgui_freq_sink_x0.set_figger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "") 384<			
369 firdes.WIN_BLACKMAN_hARRIS, #vintype 370 0, ffo 371 samp_rate, fbv 372 "", fname 373 1 fnumber of inputs 374) 375 self.qtgui_freq_sink_x_0.set_update_time(0.10) 376 self.qtgui_freq_sink_x_0.set_y_axis(-140, 10) 377 self.qtgui_freq_sink_x_0.set_y_axis(-140, 10) 378 self.qtgui_freq_sink_x_0.set_y_axis(-140, 10) 379 self.qtgui_freq_sink_x_0.set_y_axis(-140, 10) 379 self.qtgui_freq_sink_x_0.set_y_axis(-140, 10) 370 self.qtgui_freq_sink_x_0.set_y_axis(-140, 10) 371 self.qtgui_freq_sink_x_0.set_y_label('Relative Gain', 'dB') 378 self.qtgui_freq_sink_x_0.set_flage 380 self.qtgui_freq_sink_x_0.set_flage 381 self.qtgui_freq_sink_x_0.set_flage 382 self.qtgui_freq_sink_x_0.set_flage 383 self.qtgui_freq_sink_x_0.enable_control_panel(False) 384 self.qtgui_freq_sink_x_0.enable_control_panel(False) 385 if not True:	367		
370 0, ffo 371 samp_rate, fbv 372 "", fname 373 1 fnumber of inputs 374) 375 self.qtgui_freq_sink_x_0.set_update_time(0.10) 376 self.qtgui_freq_sink_x_0.set_y_axis(-140, 10) 377 self.qtgui_freq_sink_x_0.set_y_label('Relative Gain', 'dB') 378 self.qtgui_freq_sink_x_0.set_trigger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "") 379 self.qtgui_freq_sink_x_0.set_ffi_average(1.0) 382 self.qtgui_freq_sink_x_0.enable_avtoscale(True) 383 self.qtgui_freq_sink_x_0.enable_axis_labels(True) 384 self.qtgui_freq_sink_x_0.enable_control_panel(False) 385 if not True:	368		1024, #size
371 samp_rate, fbv 372 "", fname 373 1 fnumber of inputs 374 > 375 self.qtgui_freq_sink_x_0.set_update_time(0.10) 376 self.qtgui_freq_sink_x_0.setaxis(-140, 10) 377 self.qtgui_freq_sink_x_0.set_update_time(0.10) 378 self.qtgui_freq_sink_x_0.set_update_time(0.10) 379 self.qtgui_freq_sink_x_0.set_update_time(0.10) 380 self.qtgui_freq_sink_x_0.set_update_time(0.10) 381 self.qtgui_freq_sink_x_0.set_update_time(0.10) 382 self.qtgui_freq_sink_x_0.set_update_time(0.10) 383 self.qtgui_freq_sink_x_0.enable_axis_labels(True) 384 self.qtgui_freq_sink_x_0.enable_control_panel(False) 384 if not True:			
372 "", fname 373 1 fnumber of inputs 374) 375 self.qtgui_freq_sink_x_0.set_update_time(0.10) 376 self.qtgui_freq_sink_x_0.set_y_axis(-140, 10) 377 self.qtgui_freq_sink_x_0.set_y_axis(-140, 10) 378 self.qtgui_freq_sink_x_0.set_y_axis(-140, 10) 379 self.qtgui_freq_sink_x_0.set_frigger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "") 378 self.qtgui_freq_sink_x_0.set_frigger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "") 379 self.qtgui_freq_sink_x_0.set_frigger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "") 380 self.qtgui_freq_sink_x_0.set_frigger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "") 381 self.qtgui_freq_sink_x_0.set_frigger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "") 382 self.qtgui_freq_sink_x_0.set_frigger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "") 383 self.qtgui_freq_sink_x_0.set_frigger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "") 384 self.qtgui_freq_sink_x_0.set_frigger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "") 383 self.qtgui_freq_sink_x_0.set_frigger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "") 384 self.qtgui_freq_sink_x_0.set_frigger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "") 385 self.qtgui_freq_sink_x_0.set_frigger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "") 386 self.qtgui_freq_sink_x_0.set_frigger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "") 387 self.qtgui_freq_sink_x_0.set			
373 1 fnumber of inputs 374) 375 self.qtgui_freq_sink_x0.set_update_time(0.10) 376 self.qtgui_freq_sink_x0.set_y_axis(-140, 10) 377 self.qtgui_freq_sink_x0.set_y_label('Relative Gain', 'dB') 378 self.qtgui_freq_sink_x0.set_update_trigger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "") 379 self.qtgui_freq_sink_x0.enable_autoscale(False) 380 self.qtgui_freq_sink_x0.enable_grid(False) 381 self.qtgui_freq_sink_x0.enable_axis_labels(True) 382 self.qtgui_freq_sink_x0.enable_axis_labels(True) 383 self.qtgui_freq_sink_x0.enable_control_panel(False) 384 self.qtgui_freq_sink_x0.enable_control_panel(False) 385 if not True:			
374) 375 self.qtgui_freq_sink_x_0.set_update_time(0.10) 376 self.qtgui_freq_sink_x_0.set_y_axis(-140, 10) 377 self.qtgui_freq_sink_x_0.set_y_label('Relative Gain', 'dB') 378 self.qtgui_freq_sink_x_0.set_trigger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "") 379 self.qtgui_freq_sink_x_0.enable_autoscale(False) 380 self.qtgui_freq_sink_x_0.set_ffi_average(1.0) 382 self.qtgui_freq_sink_x_0.enable_axis_labels(True) 383 self.qtgui_freq_sink_x_0.enable_control_panel(False) 384 self.qtgui_freq_sink_x_0.enable_control_panel(False)			
375 self.qtgui_freq_sink_x_0.set_update_time(0.10) 376 self.qtgui_freq_sink_x_0.set_w_axis(-140, 10) 377 self.qtgui_freq_sink_x_0.set_w_axis(-140, 10) 378 self.qtgui_freq_sink_x_0.set_w_axis(-140, 10) 379 self.qtgui_freq_sink_x_0.set_wras(-140, 10) 378 self.qtgui_freq_sink_x_0.set_wras(-140, 10) 379 self.qtgui_freq_sink_x_0.set_wras(-140, 10) 380 self.qtgui_freq_sink_x_0.set_wras(-140, 10) 381 self.qtgui_freq_sink_x_0.set_wras(-160) 382 self.qtgui_freq_sink_x_0.set_wras(-10) 383 self.qtgui_freq_sink_x_0.set_wras(-10) 384 self.qtgui_freq_sink_x_0.set_wras(-10) 385 if not True:			1 snumber or inputs
376 self.qtgui_freq_sink_x_0.set_y_axis(-140, 10) 377 self.qtgui_freq_sink_x_0.set_y_label('Relative Gain', 'dB') 378 self.qtgui_freq_sink_x_0.set_trigger_mode(qtgui.TRIG_MODE_REE, 0.0, 0, "") 379 self.qtgui_freq_sink_x_0.set_grid(False) 381 self.qtgui_freq_sink_x_0.enable_avis_cale(False) 382 self.qtgui_freq_sink_x_0.enable_axis_labels(True) 383 self.qtgui_freq_sink_x_0.enable_control_panel(False) 384 self.qtgui_freq_sink_x_0.enable_control_panel(False) 385 if not True:			/ self.stgui freg sink x 0.set update time(0.10)
377 self.qtgui freq_sink_x_0.set_y_label('Relative Gain', 'dB') 378 self.qtgui freq_sink_x_0.set_trigger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "") 379 self.qtgui freq_sink_x_0.enable_autoscale(False) 380 self.qtgui freq_sink_x_0.enable_grid(False) 381 self.qtgui freq_sink_x_0.set_fft_average(1.0) 382 self.qtgui freq_sink_x_0.enable_axis_labels(True) 383 self.qtgui freq_sink_x_0.enable_control_panel(False) 384 self.qtgui freq_sink_x_0.enable_control_panel(False) 385 if not True:			
378 self.qtgui_freq_sink_x_0.set_trigger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "") 379 self.qtgui_freq_sink_x_0.enable_autoscale(False) 380 self.qtgui_freq_sink_x_0.enable_grid(False) 381 self.qtgui_freq_sink_x_0.set_fft_average(1.0) 382 self.qtgui_freq_sink_x_0.enable_autoscale(False) 383 self.qtgui_freq_sink_x_0.enable_axis_labels(True) 384 self.qtgui_freq_sink_x_0.enable_control_panel(False) 385 if not True:			
379 self.qtgui_freq_sink_x_0.enable_autoscale(False) 380 self.qtgui_freq_sink_x_0.enable_grid(False) 381 self.qtgui_freq_sink_x_0.set_fft_average(1.0) 382 self.qtgui_freq_sink_x_0.enable_axis_labels(True) 383 self.qtgui_freq_sink_x_0.enable_control_panel(False) 384 if not True:			
380 self.qtgui_freq_sink_x_0.enable_grid(False) 381 self.qtgui_freq_sink_x_0.set_fft_average(1.0) 382 self.qtgui_freq_sink_x_0.enable_axis_labels(True) 383 self.qtgui_freq_sink_x_0.enable_control_panel(False) 384 if not True:			
382 self.qtgui_freq_sink_x_0.enable_axis_labels(True) 383 self.qtgui_freq_sink_x_0.enable_control_panel(False) 384 if not True:	380		
<pre>383 self.qtgui_freq_sink_x_0.enable_control_panel(False) 384 385 if not True:</pre>	381		
384 385 if not True:			
385 if not True:			<pre>self.qtgui_freq_sink_x_0.enable_control_panel(False)</pre>
and art. ArAnt ried stur v antegrate redenally			
	500		www.dodar_traf_traf_traf_trafagate_todoue(N

```
387
                  if "complex" == "float" or "complex" == "msg float":
                    self.qtgui_freq_sink_x_0.set_plot_pos_half(not True)
                  labels = ['', '', '', '', '',
                              11, 11, 11, 11, 11j
                  widths = [1, 1, 1, 1, 1,
                             1, 1, 1, 1, 1]
                  colors = ["blue", "red", "green", "black", "cyan",
    "magenta", "yellow", "dark red", "dark green", "dark blue"]
396
                  alphas = [1.0, 1.0, 1.0, 1.0, 1.0,
398
399
                             1.0, 1.0, 1.0, 1.0, 1.0]
                  for i in xrange(1):
                      if len(labels[i]) == 0:
401
                           self.qtgui_freq_sink_x_0.set_line_label(i, "Data {0}".format(i))
                      else:
                          self.qtgui_freq_sink_x_0.set_line_label(i, labels[i])
                       self.qtgui_freq_sink_x_0.set_line_width(i, widths[i])
405
                       self.qtgui_freq_sink_x_0.set_line_color(i, colors[i])
                      self.qtgui_freq_sink_x_0.set_line_alpha(i, alphas[i])
407
                  self._qtgui_freq_sink_x_0_win = sip.wrapinstance(self.qtgui_freq_sink_x_0.pyqwidget(), Qt.QWidget)
                  self.top_grid_layout.addWidget(self._qtgui_freq_sink_x_0_win)
self.low_pass_filter_0 = filter.fir_filter_ccf(1, firdes.low_pass(
                      1, samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76))
                  self.digital_gmsk_demod_0 = digital.gmsk_demod(
                      samples per symbol=2,
                      gain_mu=0.175,
                      mu=0.5,
                      omega relative limit=0.005,
                      freq_error=0.0,
                      verbose=False,
                      log=False,
                  )
                  clr_file = open('/home/c/test_file2', 'w')
                  clr file.close()
424
                  self.blocks_multiply_const_vxx_0 = blocks.multiply_const_vcc((1, ))
                  self.blocks_file_sink_0 = blocks.file_sink(gr.sizeof_char*1, '/home/c/test_file2', False)
self.blocks_file_sink_0.set_unbuffered(False)
                   self.blks2_packet_decoder_0 = grc_blks2.packet_demod_b(grc_blks2.packet_decoder(
                           access code='',
                           threshold=-1,
                           callback=lambda ok, payload: self.blks2_packet_decoder_0.recv_pkt(ok, payload),
                      ),
                  )
                   *****
                   *****
                   self.connect((self.blks2_packet_decoder_0, 0), (self.blocks_file_sink_0, 0))
                   self.connect((self.blocks_multiply_const_vxx_0, 0), (self.low_pass_filter_0, 0))
                   self.connect((self.digital_gmsk_demod_0, 0), (self.blks2_packet_decoder_0, 0))
                  self.connect((self.low_pass_filter_0, 0), (self.digital_gmsk_demod_0, 0))
self.connect((self.low_pass_filter_0, 0), (self.qtgui_freq_sink_x_0, 0))
self.connect((self.uhd_usrp_source_0, 0), (self.blocks_multiply_const_vxx_0, 0))
              def closeEvent(self, event):
                   self.settings = Qt.QSettings("GNU Radio", "text_rx")
                   self.settings.setValue("geometry", self.saveGeometry())
                  event.accept()
454
              def get_samp_rate(self):
                  return self.samp_rate
457
              def set_samp_rate(self, samp_rate):
                   self.samp_rate = samp_rate
                   self.uhd_usrp_source_0.set_samp_rate(self.samp_rate)
                   self.qtqui freq sink x 0.set frequency range(0, self.samp rate)
```

461			self.low pass_filter_0.set_taps(firdes.low_pass(1, self.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76))	
462			Self.low_pass_filter_0.set_taps(filtdes.low_pass(f, Self.samp_rate, 200000, 50000, filtdes.win_nAmmins, 6.76))	
463 464	Ecla	88 A	ack(gr.top_block, Qt.QWidget):	
465	1010		New (group_stock) and analysis .	
466		def	<pre>finit(self):</pre>	
467			gr.top_blockinit(self, "Chat")	
468			Qt.QWidgetinit(self)	
469			self.setWindowTitle("Chat")	
470			<pre>qtgui.util.check_set_qss()</pre>	
471			try:	
472			<pre>self.setWindowIcon(Qt.QIcon.fromTheme('gnuradio-grc'))</pre>	
473			except:	
474			pass	
475			self.top_scroll_layout = Qt.QVBoxLayout()	
476			self.setLayout(self.top_scroll_layout)	
477			self.top_scroll = Qt.QScrollArea()	
478			self.top_scroll.setFrameStyle(Qt.QFrame.NoFrame)	
479 480			<pre>self.top_scroll_layout.addWidget(self.top_scroll) self.top_scroll.setWidgetResizable(True)</pre>	
481			self.top_widget = 0t.0Widget()	
482			StiftOp_scollsetWidget(self.top_widget)	
483			self.top_layout = Qt.QVBoxLayout(self.top_widget)	
484			<pre>self.top grid layout = QL.QGridLayout()</pre>	
485			self.top layout.addLayout(self.top grid layout)	
486				
487			self.settings = Qt.QSettings("GNU Radio", "chat")	
488			<pre>self.restoreGeometry(self.settings.value("geometry").toByteArray())</pre>	
489				
490				
491				
492			# Variables	
493				
494			<pre>self.samp_rate = samp_rate = 400000</pre>	
495				
496 497			<pre>weight_file = open('/home/c/chat3', 'w') weight file.write("ACK\n")</pre>	
498			weight file.close()	
499				
500				
501			# Blocks	
502				
503			self.uhd_usrp_sink_1 = uhd.usrp_sink(
504			",".join(("", "")),	
505			uhd.stream_args(
506 507			cpu_format="fo32",	
508			<pre>channels=range(1),</pre>	
509),	
510) self.uhd usrp sink 1.set samp rate(samp rate)	
511			Self.uhd_usrp_sink_!set_time_now(uhd.time_spec(time.time()), uhd.ALL_MBOARDS)	
512			self.uhd usrp sink 1.set center freq(1000000000, 0)	
513			self.uhd_usrp_sink_1.set_gain(40, 0)	
514			self.uhd_usrp_sink_1.set_antenna('TX/RX', 0)	
515			<pre>self.digital_gmak_mod_0 = digital.gmak_mod(</pre>	
516			<pre>samples_per_symbol=2,</pre>	
517			bt=0.35,	
518			verbose=False,	
519			log=False,	
520 521)	
521			<pre>self.blocks_throttle_0 = blocks.throttle(gr.sizeof_char*1, samp_rate,True) self.blocks_multiply_const_vxx_1 = blocks.multiply_const_vcc((1,))</pre>	
522			set: blocks head 0 = blocks.head(gr.sizecf that'1, 10000)	
524			self.blocks_file_source_0 = blocks.file_source(gr.sizeof_char*1, '/home/c/chat3', True)	
525			salf.blocks_file_source_0.set_begin_tag(pmt.PMT_NL)	
526			self.blks2_packet_encoder_0 = grc_blks2.packet_mod_b[grc_blks2.packet_encoder(
527			samples_per_symbol=2,	
528			<pre>bits_per_symbol=1,</pre>	
529			preamble='',	
530			access_code='',	
531			pad_for_usrp=True,	
532).	
533			payload_length=0,	
534				

535			
536			
537			
538			
539 540		# Connections	
540		self.connect((self.blks2 packet encoder 0, 0), (self.digital_gmsk_mod_0, 0))	
542		stif.connect((self.blocks_file_source_0, 0), (self.blocks_throttle_0, 0)) self.connect((self.blocks_file_source_0, 0), (self.blocks_throttle_0, 0))	
543		<pre>self.connect((self.blocks head 0, 0), (self.blks2_packet_encoder 0, 0))</pre>	
544		<pre>self.connect((self.blocks_multiply_const_vxx_1, 0), (self.uhd_usrp_sink_1, 0))</pre>	
545		<pre>self.connect((self.blocks_throttle_0, 0), (self.blocks_head_0, 0))</pre>	
546		<pre>self.connect((self.digital_gmsk_mod_0, 0), (self.blocks_multiply_const_vxx_1, 0))</pre>	
547			
548		<pre>def closeEvent(self, event):</pre>	
549 550		<pre>self.settings = Qt.QSettings("GNU Radio", "chat") self.settings.setValue("geometry", self.saveGeometry())</pre>	
551		still settings.settate(geometry , sett.savedeometry()) event.accept()	
552			
553		def get samp rate(self):	
554		return self.samp_rate	
555			
556		<pre>def set_samp_rate(self, samp_rate):</pre>	
557		self. <u>samp</u> _rate = samp_rate	
558 559		<pre>self.uhd_usrp_sink_1.set_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate)</pre>	
560		Service of the servic	
561			
562	def	ef main (options=None):	
563			
564		from distutils.version import StrictVersion	
565		if StrictVersion(Qt.qVersion()) >= StrictVersion("4.5.0"):	
566 567		<pre>style = gr.prefs().get_string('gtgui', 'style', 'raster') Ot.OApplication.setGraphicsSystem(style)</pre>	
568		gapp = Ct. OApplication (sys.argv)	
569			
570		global weight1, weight2, weight3, weight4, weight5, weight6, weight7, weight8, weight9, weight10, weight11, weight12, wei	ight13, weight14
571		a_num = 0	
572		b_num = 0	
573		c_num = 0	
573 574		c_num = 0 d_num = 0	
573 574 575		c_num = 0 d_num = 0 e_num = 0	
573 574		c_num = 0 d_num = 0	
573 574 575 576		c_num = 0 d_num = 0 e_num = 0 f_num = 0	
573 574 575 576 577 578 578		c_num = 0 d_num = 0 e_num = 0 f_num = 0 g_num = 0 h_num = 0 i_num = 0	
573 574 575 576 577 578 579 580		<pre>c_num = 0 d_num = 0 e_num = 0 f_num = 0 g_num = 0 h_num = 0 i_num = 0 j_num = 0</pre>	
573 574 575 576 577 578 579 580 581		<pre>c_num = 0 d_num = 0 e_num = 0 f_num = 0 g_num = 0 h_num = 0 i_num = 0 i_num = 0 k_num = 0</pre>	
573 574 575 576 577 578 579 580 581 582		<pre>o_num = 0 d_num = 0 e_num = 0 f_num = 0 f_num = 0 h_num = 0 h_num = 0 h_num = 0 i_num = 0 J_num = 0 l_num = 0</pre>	
573 574 575 576 577 578 579 580 581 581 582 583		<pre>c_num = 0 d_num = 0 e_num = 0 f_num = 0 g_num = 0 h_num = 0 i_num = 0 j_num = 0 j_num = 0 n_num = 0 </pre>	
573 574 575 576 577 578 579 580 581 582		<pre>o_num = 0 d_num = 0 e_num = 0 f_num = 0 f_num = 0 h_num = 0 h_num = 0 h_num = 0 i_num = 0 J_num = 0 l_num = 0</pre>	
573 574 575 576 577 578 579 580 581 582 582 583 584		<pre>c_num = 0 d_num = 0 e_num = 0 f_num = 0 g_num = 0 h_num = 0 i_num = 0 i_num = 0 k_num = 0 n_num = 0 </pre>	
573 574 575 576 577 578 579 580 581 583 583 584 583 584 585 586 587		c_num = 0 d_num = 0 e_num = 0 f_num = 0 g_num = 0 h_num = 0 i_num = 0 i_num = 0 k_num = 0 n_num = 0 n_num = 0 n_num = 0 sum_b = 0 sum_c = 0	
573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588		<pre>c_num = 0 d_num = 0 e_num = 0 f_num = 0 f_num = 0 f_num = 0 i_num = 0 i_num = 0 i_num = 0 l_num = 0 l_num = 0 a_m_num = 0</pre>	
573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 588 588 588		c_num = 0 d_num = 0 e_num = 0 f_num = 0 g_num = 0 h_num = 0 j_num = 0 j_num = 0 j_num = 0 n_num = 0 m_num = 0 sum_b = 0 sum_b = 0 sum_c = 0 sum_d = 0	
573 574 575 576 577 578 579 580 581 582 583 584 583 584 585 586 587 588 589 590		c_num = 0 d_num = 0 e_num = 0 f_num = 0 d_num = 0 h_num = 0 i_num = 0 l_num = 0 n_num = 0 n_num = 0 sum_a = 0 sum_b = 0 sum_c = 0 sum_d = 0 sum_d = 0	
573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 588 588 588		c_num = 0 d_num = 0 e_num = 0 f_num = 0 g_num = 0 h_num = 0 j_num = 0 j_num = 0 j_num = 0 n_num = 0 m_num = 0 sum_b = 0 sum_b = 0 sum_c = 0 sum_d = 0	
573 574 575 577 578 579 580 581 582 583 584 583 584 585 586 587 588 589 590 591		c_num = 0 d_num = 0 e_num = 0 f_num = 0 g_num = 0 h_num = 0 i_num = 0 k_num = 0 h_num = 0 n_num = 0 n_num = 0 sum_a = 0 sum_b = 0 sum_b = 0 sum_b = 0 sum_f = 0 sum_g = 0	
573 574 575 577 578 579 580 581 582 583 584 585 586 585 586 589 590 591 592 593 594		c_num = 0 d_num = 0 e_num = 0 f_num = 0 f_num = 0 i_num = 0 i_num = 0 l_num = 0 n_num = 0 n_num = 0 sum_a = 0 sum_b = 0 sum_c = 0 sum_d = 0 sum_f = 0 sum_f = 0 sum_f = 0 sum_f = 0 sum_f = 0	
573 574 575 576 577 580 581 582 583 584 583 584 585 588 588 589 590 591 592 591 592 593 593		<pre>o_num = 0 d_num = 0 f_num = 0 f_num = 0 f_num = 0 f_num = 0 h_num = 0 h_num = 0 h_num = 0 l_num = 0 l_num = 0 m_num = 0 sum_a = 0 sum_b = 0 sum_b = 0 sum_c = 0 sum_f = 0 sum_f = 0 sum_i = 0</pre>	
573 574 576 577 579 580 581 583 584 583 584 585 586 589 590 591 592 593 594 595 595		<pre>c_num = 0 d_num = 0 e_num = 0 f_num = 0 f_num = 0 h_num = 0 h_num = 0 i_num = 0 i_num = 0 1_num = 0 n_num = 0 sum_b = 0 sum_b = 0 sum_c = 0 sum_c = 0 sum_f = 0 sum_f = 0 sum_i = 0 s</pre>	
573 574 576 577 580 581 582 583 584 583 584 585 586 587 588 589 590 591 592 593 594 595 595 595 597		<pre>c_num = 0 d_num = 0 f_num = 0 f</pre>	
573 574 575 576 577 580 581 582 583 584 583 584 585 586 587 588 589 590 591 592 593 594 595 595 595		<pre>c_num = 0 d_num = 0 e_num = 0 f_num = 0 f</pre>	
573 574 576 577 579 580 581 583 584 583 584 585 586 589 590 591 592 593 594 595 595 595 595 596 597 598 599		<pre>o_num = 0 d_num = 0 f_num = 0 f_num = 0 f_num = 0 f_num = 0 i_num = 0 i_num = 0 i_num = 0 i_num = 0 n_num = 0 sum_b = 0 sum_b = 0 sum_c = 0 sum_f = 0 sum_f = 0 sum_i = 0 s</pre>	
573 574 575 576 577 580 581 582 583 584 583 584 585 586 587 588 589 590 591 592 593 594 595 595 595		<pre>c_num = 0 d_num = 0 e_num = 0 f_num = 0 f</pre>	
573 574 575 576 577 580 581 582 583 584 585 586 586 589 590 591 592 593 594 593 594 595 595 595 595 595 595 599 600		<pre>c_num = 0 d_num = 0 f_num = 0 f_num = 0 f_num = 0 f_num = 0 i_num = 0 i_num = 0 i_num = 0 i_num = 0 n_num = 0 n_num = 0 sum_a = 0 sum_d = 0 sum_f = 0 sum_i = 0 s</pre>	
573 574 575 576 579 580 581 583 583 584 585 586 589 590 591 593 594 595 593 594 595 595 595 595 595 595 595 599 600 601 602 603		<pre>o_num = 0 d_num = 0 d_num = 0 f_num = 0 f</pre>	
573 574 575 576 577 580 581 582 583 584 583 584 585 586 587 588 590 591 593 594 595 595 596 597 598 599 595 596 597 598 599 600 601 602 603 604		<pre>o_num = 0 d_uum = 0 d_uum = 0 f_num = 0 f</pre>	
573 574 575 576 579 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594 595 595 595 595 595 596 597 598 599 600 602 603 602 603 604 605		o_num = 0 d_num = 0 e_num = 0 f_num = 0 i_num = 0 i_num = 0 i_num = 0 n_num = 0 n_num = 0 n_num = 0 a_num = 0 sum_b = 0 sum_c = 0 sum_c = 0 sum_f = 0 sum_n = 0 sum_i = 0 sum_n = 0 sum_n = 0 sum_n = 0 sum_n = 0 sum_i = 0	
573 574 575 576 577 579 580 581 583 584 583 584 585 586 589 590 591 592 593 594 595 595 595 596 597 598 599 600 601 602 603 604 605 605 605		o_num = 0	
573 574 575 576 579 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594 595 595 595 595 595 596 597 598 599 600 602 603 602 603 604 605		o_num = 0 d_num = 0 e_num = 0 f_num = 0 i_num = 0 i_num = 0 i_num = 0 n_num = 0 n_num = 0 n_num = 0 a_num = 0 sum_b = 0 sum_c = 0 sum_c = 0 sum_f = 0 sum_n = 0 sum_i = 0 sum_n = 0 sum_n = 0 sum_n = 0 sum_n = 0 sum_i = 0	

609	predict $k = 0$
610	predict 1 = 0
611	predict m = 0
612	predict n = 0
613	actual_alist = []
614	actual_list = []
615	
	actual_c_list = []
616	actual_d_list = []
617	actual_e_list = []
618	actual_f_list = []
619	actual_g_list = []
620	actual_h_list = []
621	actual_i_list = []
622	actual_j_list = []
623	<pre>actual_k_list = []</pre>
624	actual_l_list = []
625	actual_m_list = []
626	actual_n_list = []
627	<pre>predict_a_list = []</pre>
628	predict_b_list = []
629	predict_c_list = []
630	predict_d_list = []
631	predict_e_list = []
632	<pre>predict_f_list = []</pre>
633	<pre>predict_g_list = []</pre>
634	<pre>predict_h_list = []</pre>
635	predict_i_list = []
636	<pre>predict_j_list = []</pre>
637	<pre>predict_k_list = []</pre>
638	predict_l_list = []
639	predict_m_list = []
640	predict_n_list = []
641	
642	tb = chat()
643	<pre>tb.start()</pre>
644	<pre> time.sleep(1) </pre>
645	tb.stop()
646	
647	<pre>final = open('/home/c/final', 'w')</pre>
648	final.close()
649	
650 🖯	while True:
651	tb = text_rx()
652	tb.start()
653	
654	<pre>data1 = open('/home/c/test_file2', 'r+')</pre>
655	<pre>final = open('/home/c/final', 'a')</pre>
656	
657 🤤	while True:
658	data1.flush()
659	data1.readline()
660	<pre>file_content2 = data1.readline()</pre>
661 🤤	<pre>if (file_content2 == "eof\n"):</pre>
662	<pre>print(file_content2)</pre>
663 🗎	break
664 🗦	<pre>elif (file_content2 != ""):</pre>
665	<pre>final.write(file_content2)</pre>
666	<pre>split = file_content2.split(' ')</pre>
667 🖯	<pre>if (split[1].startswith('10') and (split[1].startswith('0', 2) == False)):</pre>
668	a_num += 1
668 669	<pre>a_num += 1 actual_a_list.append(float(split[0]))</pre>
669	actual_a_list.append(float(split[0]))
669 670	<pre>actual_a_list.append(float(split[0])) sum_a += float(split[0])</pre>
669 670 671	<pre>actual_a_list.append(float(split[0])) sum_a += float(split[0]) predict_a = sum_a/a_num</pre>
669 670 671 672	<pre>actual_a_list.append(float(split[0])) sum_a += float(split[0]) predict_a == sum_a/a_num print("Prediction for Next A: "+str(predict_a))</pre>
669 670 671 672 673	<pre>actual_a_list.append(float(split[0])) sum_a += float(split[0]) predict_a = sum_a/a_num print("Prediction for Next A: "+str(predict_a)) predict_a_list.append(predict_a)</pre>
669 670 671 672 673 ⊖ 674 ⊽	<pre>actual_a_list.append(float(split[0])) sum_a += float(split[0]) predict_a = sum_a/a_num print("Prediction for Next A: "+str(predict_a)) predict_a_list.append(predict_a) elif(split[1].startswith('20')):</pre>
669 670 671 672 673 674 675	<pre>actual_a_list.append(float(split[0])) sum_a += float(split[0]) predict_a = sum_a/a_num print("Prediction for Next A: "+str(predict_a)) predict_a_list.append(predict_a) ellif(split[1].startswith('20')): b_num += 1</pre>
669 671 672 673 674 675 676	<pre>actual_a_list.append(float(split[0])) sum_a += float(split[0]) predict_a = sum_a/a_num print("Prediction for Next A: "+str(predict_a)) predict_a_list.append(predict_a) ellf (split[1].startswith('20')): b_num += 1 actual_b_list.append(float(split[0]))</pre>
669 670 671 672 673 674 675 675 676 677	<pre>actual_a_list.append(float(split[0])) sum_a += float(split[0]) predict_a = sum_a/a_num print("Prediction for Next A: "+str(predict_a)) predict_a_list.append(predict_a) elif (split[1].startswith('20')): b_num += 1 actual_b_list.append(float(split[0])) sum_b += float(split[0])</pre>
669 670 671 672 673 674 675 675 676 677 678	<pre>actual_a_list.append(float(split[0])) sum_a += float(split[0]) predict_a = sum_a/a_num print("Prediction for Next A: "+str(predict_a)) predict_a_list.append(predict_a) ellif (split[1].startswith('20')): b_num += 1 actual_b_list.append(float(split[0])) sum_b += float(split[0]) predict_b = sum_b/b_num</pre>
669 670 671 672 673 674 675 676 677 678 679	<pre>actual_a_list.append(float(split[0])) sum_a += float(split[0]) predict_a = sum_a/a_num print("Prediction for Next A: "+str(predict_a)) predict_a_list.append(predict_a) ellif(split[1].startswith('20')): b_num += 1 actual_b_list.append(float(split[0])) sum_b += float(split[0]) predict_b = sum_b/b_num print("Prediction for Next B: "+str(predict_b))</pre>

683	actual_c_list.append(float(split[0]))
684	<pre>sum_c += float(split[0])</pre>
685	predict_c = sum_c/c_num
686	<pre>print("Prediction for Next C: "+str(predict_c))</pre>
687	predict_c_list.append(predict_c)
688	<pre>elif (split[1].startswith('40')):</pre>
689	d_num += 1
690	<pre>actual_d_list.append(float(split[0])) rum_d_f_list_append(float(split[0]))</pre>
691 692	<pre>sum_d += float(aplit(0)) provided = cum_d(d cum_d)</pre>
693	<pre>predict_d = sum_d/d_num print("Prediction for Next D: "+str(predict_d))</pre>
694	predict_ellist.append(predict_d)
695	elif (split]].startswith('50')):
696	e num += 1
697	
698	<pre>sum_e += float(split[0])</pre>
699	predict_e = sum_e/e_num
700	<pre>print("Prediction for Next E: "+str(predict_e))</pre>
701	<pre>predict_e_list.append(predict_e)</pre>
702	<pre>elif (split[1].startswith('60')):</pre>
703	f_num += 1
704	actual_f_list.append(float(split[0]))
705 706	<pre>sum_f += float(split[0]) predict_f = sum_f/f_num</pre>
707	print("Prediction for Next F: "+str(predict_f))
708	predict_f_list.append(predict_f)
709	<pre>elif (split[1].startswith('70')):</pre>
710	g_n um $\neq 1$
711	actual_g_list.append(float(split[0]))
712	<pre>sum_g += float(split[0])</pre>
713	predict_g = sum_g/g_num
714	<pre>print("Prediction for Next G: "+str(predict_g))</pre>
715	<pre>predict_g_list.append(predict_g)</pre>
716	<pre>elif (split[1].startswith('80')):</pre>
717	$h_{num} += 1$
718	actual_h_list.append(float(split[0])) sum h += float(split[0])
720	predict h = sum h/h num
721	print("Prediction for Next H: "+str(predict_h))
722	predict h list.append(predict h)
723	<pre>elif (split[1].startswith('90')):</pre>
724	i_num += 1
725	actual_i_list.append(float(split[0]))
726	<pre>sum_i += float(split[0])</pre>
727	predict_i = sum_i/i_num
728	<pre>print("Prediction for Next I: "+str(predict_i))</pre>
729 730	predict i list.append(predict i)
731	<pre>elif (split[1].startswith('100')):</pre>
732	actual_j_list.append(float(split[0]))
733	<pre>sum j += float(split[0])</pre>
734	predict j = sum j/j num
735	print("Prediction for Next J: "+str(predict_j))
736	<pre>predict_j_list.append(predict_j)</pre>
737	<pre>elif (split[1].startswith('110')):</pre>
738	k_num += 1
739	<pre>actual_k_list.append(float(split[0]))</pre>
739 740	<pre>actual_k_list.append(float(split[0])) sum_k += float(split[0])</pre>
739 740 741	<pre>a-tual_k_list.append(float(split[0])) sum_k += float(split[0]) predict_k = sum_k/k_num</pre>
739 740 741 742	<pre>actual_k_list.append(float(split[0])) sum_k += float(split[0]) predict_k = sum_k/k_num print("Prediction for Next K: "+str(predict_k))</pre>
739 740 741	<pre>actual_k_list.append(float(split[0])) sum_k += float(split[0]) predict_k = sum_k/k_num print("predicting for Next K: "+str(predict_k)) predict_k_list.append(predict_k)</pre>
739 740 741 742 743	<pre>actual_k_list.append(float(split[0])) sum_k += float(split[0]) predict_k = sum_k/k_num print("Prediction for Next K: "+str(predict_k))</pre>
739 740 741 742 743 744	<pre>actual_k_list.append(float(split[0])) sum_k += float(split[0]) predict_k = sum_k/k_num print("Prediction for Next K: "+str(predict_k)) predict_k_list.append(predict_k) elif (split[1].startswith('120')):</pre>
739 740 741 742 743 744 745	<pre>actual_k_list.append(float(split[0])) sum_k += float(split[0]) predict_k = sum_k/k_num print("Prediction for Next K: "+str(predict_k)) predict_k_list.append(predict_k) elif (split[1].startswith('120')): l_num += 1</pre>
739 740 741 742 743 744 745 746	<pre>actual_k_list.append(float(split[0])) sum_k += float(split[0]) predict_k = sum_k/k_num print("Prediction for Next K: "+str(predict_k)) predict_k_list.append(predict_k) elif(split[1].startswith('120')): l_num += 1 actual_l_list.append(float(split[0]))</pre>
739 740 741 742 743 744 745 746 747 748 749	<pre>a_tual_k_list.append(float(split[0])) sum_k += float(split[0]) predict_k = sum_k/k_num print("PredictInf for Next K: "+str(predict_k)) predict_k_list.append(predict_k) elif(split[1].startswith('120')):</pre>
739 740 741 742 743 744 745 746 747 748 749 750	<pre>artual_k_list.append(float(split[0])) sum_k += float(split[0]) predict_k = sum_k/k_num print("Prediction for Next K: "+str(predict_k)) predict_k_list.append(predict_k) elif(split[1].startswith('120')): l_num += 1 actual_l_list.append(float(split[0])) sum_l += float(split[0]) predict_l = sum_l/l_num print("Prediction for Next L: "+str(predict_l)) predict_l_list.append(predict_l)</pre>
739 740 741 742 743 744 745 746 747 748 749 749 750 751	<pre>artual_k_list.append(float(split[0])) sum_k += float(split[0]) predict_k = sum_k/k_num print("Prediction for Next K: "+str(predict_k)) predict_k_list.append(predict_k) elif(split[1].startswith('120')): l_num += 1 actual_list.append(float(split[0])) sum_1 += float(split[0]) predict_l = sum_l/l_num print("Prediction for Next L: "+str(predict_l)) predict_l_list.append(predict_l) elif(split[1].startswith('130')):</pre>
739 740 741 742 743 744 745 746 747 748 749 749 750 751 752	<pre>artual_k_list.append(float(split[0])) sum_k += float(split[0]) predict_k = sum_k k_num print("Prediction for Next K: "+str(predict_k)) predict_k_list.append(predict_k) elif (split[1].startswith('120')): 1_num += 1 actual_l_list.append(float(split[0])) sum_l += float(split[0]) predict_l = sum_l/1_num print("Prediction for Next L: "+str(predict_l)) predict_l_list.append(predict_l) elif (split[1].startswith('130')): m_num += 1</pre>
739 740 741 742 743 744 745 746 747 748 749 750 751 752 753	<pre>artual_k_list.append(float(split[0])) sum_k += float(split[0]) predict_k = sum_k/k_num print("Prediction for Next K: "+str(predict_k)) predict_k_list.append(predict_k) elif(split[1].startswith('120')):</pre>
739 740 741 742 743 745 746 747 746 747 748 749 750 751 752 753 754	<pre>artual_k_list.append(float(split[0])) sum_k += float(split[0]) predict_k = sum_k/k_num print("Prediction for Next K: "+str(predict_k)) predict_k_list.append(predict_k) elif(split[1].startswith('120')): l_num += 1 actual_l_list.append(float(split[0])) sum_1 += float(split[0]) predict_l = sum_l/l_num print("Prediction for Next L: "+str(predict_l)) predict_l_list.append(predict_l) elif(split[1].startswith('130')): m_num += 1 actual_m_list.append(float(split[0])) sum_m += float(split[0])) </pre>
739 740 741 742 743 745 746 747 746 747 748 749 750 751 752 753 754 755	<pre>artual_k_list.append(float(split[0])) sum_k += float(split[0]) predict_k = sum_k k_num print("Prediction for Next K: "+str(predict_k)) predict_k_list.append(predict_k) elif (split[1].startswith('l20')): l_num += 1 actual_l_list.append(float(split[0])) sum_1 += float(split[0]) predict_l = sum_l/l_num print("Prediction for Next L: "+str(predict_l)) predict_l = sum_l/n_num </pre>
739 740 741 742 743 744 745 746 747 746 747 748 749 750 751 752 753 754	<pre>artual_k_list.append(float(split[0])) sum_k += float(split[0]) predict_k = sum_k/k_num print("Prediction for Next K: "+str(predict_k)) predict_k_list.append(predict_k) elif(split[1].startswith('120')): l_num += 1 actual_l_list.append(float(split[0])) sum_1 += float(split[0]) predict_l = sum_l/l_num print("Prediction for Next L: "+str(predict_l)) predict_l_list.append(predict_l) elif(split[1].startswith('130')): m_num += 1 actual_m_list.append(float(split[0])) sum_m += float(split[0])) </pre>

757	predict m_list.append(predict m)
758	<pre>elif (split[1].startswith('140')):</pre>
759	n num $+= 1$
760	actual_n_list.append(float(split[0]))
761	$sum_n + = float(split(0))$
762	predict n = sum n/n num
763	print("Prediction for Next N: "+str(predict_n))
764	predict_n_list.append(predict_n)
765	<pre>print("Current Value: "+file_content2)</pre>
766	time.sleep(1)
767	sb = ack()
768	sb.start()
769	print, "SENDING ACK"
770	time.sleep(1)
771	sb.stop()
772	break
773	time.sleep(1)
774	
775	tb.stop()
776	
777	<pre>if (file_content2 == "eof\n"):</pre>
778	data1.close()
779	final.close()
780	break
781	
782	data1.close()
783	final.close()
784	
785	tb.stop()
786	
787	sum_sq_a = 0
788	for z in range(0, len(predict a list)-1):
789	square = (actual a list[z1] - predict a list[z])**2
790	
	sum_sq_a += square
791	<pre>msqrterr_a = math.sqrt(sum_sq_a)</pre>
792	print("Root Mean Squared Error for Sensor 1: "+str(msqrterr_a))
793 794	$s_{\rm un} = 0$
795	<pre>for z in range(0, len(predict_b_list)-1):</pre>
796	<pre>square = (actual_b_list[z+1] - predict_b_list[z])**2</pre>
797	sum_sq_b += square
798	<pre>msgrterr_b = math.sqrt(sum_sq_b)</pre>
799	print("Root Mean Squared Error for Sensor 2: "+str(msqrterr_b))
800	
801	sum_sq_c = 0
802	<pre>for z in range(0, len(predict c list)-1):</pre>
803	<pre>square = (actual_c list[z+1] - predict c list[z])**2</pre>
804	sum sq_c += square
805	msqrter_c = math.sqrt(sum_sq_c)
806	print("Root Mean Squared Error for Sensor 3: "+str(msqrterr_c))
807	• • • • • • • • • • • • • • • • • • • •
808	0 = b pe mue
809	sum sq. d - 0 for z in range(0, len(predict d list)-1):
810	
	<pre>square = (actual_d_list[z+1] - predict_d_list[z])**2 square = d = square d = square</pre>
811	sum_sq_d += square
812	msqrterr_d = math.sqrt(sum_sq_d)
813	print("Root Mean Squared Error for Sensor 4: "+str(msqrterr_d))
814	
815	sum_sq_e = 0
816	<pre>for z in range(0, len(predict_e_list)-1):</pre>
817	<pre>square = (actual_e_list[z+1] - predict_e_list[z])**2</pre>
818	sum_sq_e += square
819	<pre>msqrterr_e = math.sqrt(sum_sq_e)</pre>
820	print("Root Mean Squared Error for Sensor 5: "+str(msqrterr_e))
821	-
822	sum_sq_f = 0
823	for z in range(0, len(predict_f_list)-1):
824	square = (actual_f_list[z1] - predict_f_list[z])**2
825	sumse_[+= square
826	sum_sqL + - square msqterr f = math.sqt(sum sq f)
827	print("Root Mean Squared Error for Sensor 6: "+str(msqrterr_f))
828	
829	sum_sq_g = 0
830	<pre>for z in range(0, len(predict_g_list)-1):</pre>

831		square = (actual_g_list[z+1] - predict_g_list[z])**2
832		sum_sq_g += square
833		msqrterr g = math.sqrt(sum_sq_g)
834		print("Root Mean Squared Error for Sensor 7: "+str(msqrterr_g))
835		
836		sum sq h = 0
837		for z in range(0, len(predict h list)-1):
838		square = (actual h list[z]] - prodict h list[z])**2
839		sum_sq_h += square
840		<pre>msqrterr_h = math.sqrt(sum_sq_h)</pre>
841		print("Root Mean Squared Error for Sensor 8: "+str(msqrterr_h))
842		
843		sum_sq_i = 0
844		<pre>for z in range(0, len(predict_i_list)-1):</pre>
845		<pre>square = (actual_i_list[z+1] - predict_i_list[z])**2</pre>
846		sum_sq_i += square
847		msgrterr_i = math.sqrt(sum_sq_i)
848		print("Root Mean Squared Error for Sensor 9: "+str(msqrterr_i))
849		
850		sum_sq_j = 0
851		
852		for z in range(0, len(predict j list)-1):
		<pre>square = (actual j_list[z+1] - predict_j_list[z])**2</pre>
853		sum_sq_j += square
854		<pre>msqrterr_j = math.sqrt(sum_sq_j)</pre>
855		print("Root Mean Squared Error for Sensor 10: "+str(msqrterr_j))
856		
857		sum_sq_k = 0
858		<pre>for z in range(0, len(predict_k_list)-1):</pre>
859		<pre>square = (actual_k_list[z+1] - predict_k_list[z])**2</pre>
860		sum sq k += square
861		msqrteir_k = math.sqrt(sum_sq_k)
862		print("Root Mean Squared Error for Sensor 11: "+str(msqrterr k))
863		Fine (1000 1941 Starter File Senser 11, 1001 (medicorr_v))
864		
		sum_sql = 0
865		<pre>for z in range(0, len(predict_l_list)-1):</pre>
866		<pre>square = (actual_list[z+1] - predict_list[z])**2</pre>
867		sum_sq_1 += square
868		<pre>msqrterr_1 = math.sqrt(sum_sq_1)</pre>
869		print("Root Mean Squared Error for Sensor 12: "+str(msgrterr_1))
870		
871		sum_sq_m = 0
872		<pre>for z in range(0, len(predict_m_list)-1):</pre>
873		<pre>square = (actual_m_list[z+1] - predict_m_list[z])**2</pre>
874		sum_sq_m += square
875		msgrterr_m = math.sqrt(sum_sq_m)
876		print("Root Mean Squared Error for Sensor 13: "+str(msqrterr_m))
877		······································
878		sum_sq_n = 0
878		
		for z in range(0, len(predict <u>n</u> list)-1):
880		<pre>square = (actual_n_list[z+1] - predict_n_list[z])**2</pre>
881		sum_sq_n += square
882		<pre>msqrterr_n = math.sqrt(sum_sq_n)</pre>
883		print("Root Mean Squared Error for Sensor 14: "+str(msqrterr_n))
884		
885		rmse_tot = msqrterr_a+msqrterr_b+msqrterr_c+msqrterr_d+msqrterr_f+msqrterr_g+msqrterr_h+msqrterr_i+msqrterr_j+msqrterr_k+msqrterr_
886		print("\nTotal Root Mean Squared Error: "+str(rmse_tot))
887		
888		exit()
889		
890		def quitting():
891		tb.stop()
892		tb.wait()
893		<pre>qapp.connect(qapp, Qt.SIGNAL("aboutToQuit()"), quitting)</pre>
894		dapp.exec_()
895		
896		
896 897 🕨	if	name == 'main':
896	if	name == 'main': main()

Figure A.1: Weighted Round-Robin Algorithm for Base Station Radio

1		isr/bin/env python2			
2	<pre># -*- coding: utf-8 -*- ###################################</pre>				
4		W Radio Python Flow Graph			
5		tile: Chat2			
6	# G	nerated: Thu Feb 10 09:52:14 2022			
7	0 ###;				
8					
9 🕨	9if .				
10		import ctypes			
11		<pre>import sys if sys.platform.startswith('linux'):</pre>			
13		try:			
14		x11 = ctypes.cdll.LoadLibrary('libX11.so')			
15		x11.XInitThreads()			
16		except:			
17		<pre>print_"Warning: failed to XInitThreads()"</pre>			
18					
19 39	LURS	Stand -			
40					
41	clas	s chat2(gr.top block, Qt.QWidget):			
42					
43		definit(self):			
44		gr.top_blockinit(self, "Chat2")			
45 46		Qt.QWidgetinit(self) self.setWindowTitle("Chat2")			
47		self.secandowille(\under) qtqui.util.check_set_qss()			
48		try:			
49		<pre>self.setWindowIcon(Qt.QIcon.fromTheme('gnuradio-gra'))</pre>			
50		except:			
51		pass			
52		self.top_scroll_layout = Qt.QVBoxLayout()			
53 54		<pre>self.setLayout(self.top_scroll_layout) self.top_scroll = Qt.QScrollArea()</pre>			
55		self.top_sorbit = (C.Sorbithied) self.top_sorbit.estramstyle(Qt.CFrame.NoFrame)			
56		self.top scroll layout.addWidget(self.top scroll)			
57		self.top_scroll.setWidgetResizable(True)			
58		<pre>self.top_widget = Qt.QWidget()</pre>			
59		self.top_scoll.setWidget(self.top_widget)			
60 61		<pre>self.top_layout = Qt.QVBoxLayout(self.top_widget) self.top_grid_layout = Qt.QGridLayout()</pre>			
62		self.top_layout.addLayout(self.top_grid_layout)			
63					
64		self.settings = Qt.QSettings("GNU Radio", "chat2")			
65		<pre>self.restoreGeometry(self.settings.value("geometry").toByteArray())</pre>			
66 67					
68					
69		# Variables			
70					
71		<pre>self.gamp_rate = gamp_rate = 400000</pre>			
72					
73 74		######################################			
74		# blocks			
76		self.uhd_usrp_source_0 = uhd.usrp_source(
77		","-join((", "")),			
78		uhd.stream_args(
79		cpu_format="fo32",			
80		channels=range(1),			
81 82),			
83) self.uhd usrp source 0.set samp rate(samp rate)			
84		self.uhd_usrp_source_0.set_center_freq(100000000, 0)			
85		<pre>self.uhd_usrp_source_0.set_gain(30, 0)</pre>			
86		<pre>self.uhd_usrp_source_0.set_antenna('TX/RX', 0)</pre>			
87		<pre>self.gtgui_freq_sink_x_0 = qtgui.freq_sink_c(</pre>			
88		1024, frize			
89 90		<pre>firdes.WIN_BLACKMAN_hARRIS, #vintype 0, #fc</pre>			
91		samprate, #bv			
92		", frame			
93		1 #number of inputs			

94		
96	self.qtgui freq sink x 0.set update time(0.10)	
95	self.qtgui_freq_sink_x_0.set_y_axis(-140, 10)	
98	<pre>self.qtgui_freq_sink_x_0.set_y_label('Relative Gain', 'dB') self.qtgui freq_sink_x_0.set_trigger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "")</pre>	
99	self.qgui_freq_sink_x 0.enable_autoscale(False)	
100	self.qtqui freq sink x 0.enable qrid(False)	
101	self.qtgui freq sink x 0.set fft average(1.0)	
102	<pre>self.qtgui_freq_sink_x_0.enable_axis_labels(True)</pre>	
103	<pre>self.qtgui_freq_sink_x_0.enable_control_panel(False)</pre>	
104		
105	if not True:	
106	<pre>self.qtgui_freq_sink_x_0.disable_legend()</pre>	
107	if "complex" == "float" or "complex" == "msg float":	
108	<pre>if "complex" "float" or "complex" "mag_float": self.qtgui freq sink x 0.est plot pos half(not True)</pre>	
110	Serr. degut_rred_stik_A_0.set_brot_pos_mart(not_free)	
111 🕀	labels = ['', '', '', '', '',	
112		
113 🤤	widths = $[1, 1, 1, 1, 1, 1]$	
114 🖨	1, 1, 1, 1]	
115 🤤	colors = ["blue", "red", "green", "black", "cyan",	
116	"magenta", "yellow", "dark red", "dark green", "dark blue"]	
117 👤	alphas = [1.0, 1.0, 1.0, 1.0, 1.0, 1.0,	
118 A 119 D	1.0, 1.0, 1.0, 1.0, 1.0]	
119	<pre>for i in xrange(1): if len(labels[i]) == 0:</pre>	
120	<pre>ii ien(tabels[1]) 0: self.qtyui freq_sink x_0.set_line_label(i, "Data (0)".format(i))</pre>	
122	else:	
123	self.qtgui_freq_sink_x_0.set_line_label(i, labels[i])	
124	self.qtgui freq sink x 0.set line width(i, widths[i])	
125	self.qtgui_freq_sink_x_0.set_line_color(i, colors[i])	
126	<pre>self.qtgui_freq_sink_x_0.set_line_alpha(i, alphas[i])</pre>	
127		
128	<pre>selfqtgui_freq_sink_x_0_win = sip.wrapinstance(self.qtgui_freq_sink_x_0.pyqwidget(), Qt.QWidget)</pre>	
129	<pre>self.top_grid_layout.addWidget(selfgtgui_freq_sink_x_0_win) self.low pass filter 0 = filter.fir filter cof(1, firdes.low pass(</pre>	
130	self low pass filter 0 - filter.fif filter cor(1, firdes.low pass(1, samp rate, 200000, 5000, firdes.Wir HAMMING (6.76))	
132	self.digital_gmsk_demod_1 = digital.gmsk_demod(
133	<pre>samples_per_symbol=2,</pre>	
134	gain_mu=0.175,	
135	mu=0.5,	
136	<pre>cmega_relative_limit=0.005,</pre>	
137	freq_error=0.0,	
138	verbose=False, log=False,	
140	log-raise,	
141	, self.blocks_multiply_const_vxx_1 = blocks.multiply_const_vcc((1,))	
142	<pre>self.blocks_file_sink_1 = blocks.file_sink(gr.sizeof_char*1, '/home/b/chat_test1.txt', False)</pre>	
143	<pre>self.blocks_file_sink_1.set_unbuffered(False)</pre>	
144	<pre>self.blks2_packet_decoder_1 = grc_blks2.packet_demod_b(grc_blks2.packet_decoder(</pre>	
145	access_code='',	
146	threshold=-1,	
147	callback=lambda ok, payload: self.blks2_packet_decoder_1.recv_pkt(ok, payload),	
148),	
150	,	
151		
152		
153 🗢		
154	# Connections	
155		
156	<pre>self.connect((self.blks2_packet_decoder_1, 0), (self.blocks_file_sink_1, 0))</pre>	
157	<pre>self.connect((self.blocks_multiply_const_vxx1, 0), (self.low_pass_filter_0, 0))</pre>	
158	<pre>self.connect((self.digital_gmsk_demod_1, 0), (self.blks2_packet_decoder_1, 0)) self.connect((self.low_pass_filter_0, 0), (self.digital_gmsk_demod_1, 0))</pre>	
160	<pre>self.connet((self.low_pass_inter_0, 0), (self.dugitar_gms_lemod_r, 0)) self.connet((self.low_pass_filter 0, 0), (self.dugitar_teq_sink x 0, 0))</pre>	
161	self.connect((self.uhd_usrp_source), 0), (self.blocks.multiply_const_vxx1, 0))	
162		
163 🖯 🖉	<pre>def closeEvent(self, event):</pre>	
164	<pre>self.settings = Qt.QSettings("GNU Radio", "chat2")</pre>	
165	<pre>self.settings.setValue("geometry", self.saveGeometry())</pre>	
166 -	event.accept()	

```
168
             def get_samp_rate(self):
                 return self.samp rate
             def set_samp_rate(self, samp_rate):
                 self.samp_rate = samp_rate
                 self.uhd_usrp_source_0.set_samp_rate(self.samp_rate)
                 self.qtgui_freq_sink_x_0.set_frequency_range(0, self.samp_rate)
                 self.low_pass_filter_0.set_taps(firdes.low_pass(1, self.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76))
176
        class text tx(gr.top block):
179
             def __init__(self):
                gr.top_block.__init__(self, "Text Tx")
                 *****
                 # Variables
                self.samp_rate = samp_rate = 400000
                 source =
                 # Block
                 ****
                 self.uhd_usrp_sink_0 = uhd.usrp_sink(
                    ",".join(("", "")),
                    uhd.stream_args(
                        cpu format="fc32",
196
                        channels=range(1),
                    5.
                 self.uhd_usrp_sink_0.set_samp_rate(samp_rate)
                 self.uhd_usrp_sink_0.set_time_now(uhd.time_spec(time.time()), uhd.ALL_MBOARDS)
                 self.uhd_usrp_sink_0.set_center_freq(1000000000, 0)
                self.uhd_usrp_sink_0.set_gain(40, 0)
self.uhd_usrp_sink_0.set_antenna('TX/RX', 0)
self.digital_gmsk_mod 0 = digital.gmsk_mod(
    samples_per_symbol=2,
                    bt=0.35,
                    verbose=False
                    log=False,
                self.blocks_throttle_0 = blocks.throttle(gr.sizeof_char*1, samp_rate,True)
                self.blocks_multiply_const_vxx_0 = blocks.multiply_const_vcc((1, ))
self.blocks_head_0 = blocks.head(gr.sizeof_char*1, samp_rate/4)
                 send = open('/home/b/tx.txt', 'w')
                if switch == 'a':
                    send.write(data0[a1])
                    print(data0[a1])
                 elif switch == 'b':
                    send.write(data1[b1])
                    print(data1[b1])
                elif switch == 'c':
                   send.write(data2[c1])
                    print(data2[c1])
                elif switch == 'd':
                    send.write(data3[d1])
                    print(data3[d1])
                 elif switch == 'e':
                    send.write(data4[e1])
                    print(data4[e1])
                 elif switch == 'f':
                    send.write(data5[f1])
                    print(data5[f1])
                 elif switch == 'g':
                    send.write(data6[g1])
                print(data6[g1])
elif switch == 'h':
                    send.write(data7[h1])
                    print(data7[h1])
                 elif switch == 'i':
                     send.write(data8[i1])
                    print(data8[i1])
```

242		elif switch == 'j':	
243		<pre>send.write(data9[j1])</pre>	
244		<pre>print(data9[j1])</pre>	
245		elif switch == 'k':	
246		send.write(data10[k1])	
247		print(data10[k1])	
248		elif switch == 'l':	
249		send.write(data11[11])	
250		print(data11[1])	
251		elif switch == 'm':	
252		<pre>sed.write(data12[m1]) </pre>	
253		print(data12[m1])	
254		elif switch $== n'$:	
255		<pre>send.write(data13[n1])</pre>	
256 257		<pre>print(data13[n1]) elif switch == 'eof':</pre>	
258		ell situe eel : send.write("eofh")	
250		send write ("ed.(m") print, "eof"	
260		sent.close()	
261		Send. (10Sel)	
262		<pre>self.blocks_file_source_0 = blocks.file_source(gr.sizeof_char*1, '/home/b/tx.txt', True)</pre>	
263			
264		self.blocks file source_0.set begin tag(pmt.FMT NIL)	
265		self. <u>blks2_packet_encoder_0 = grc_blks2.packet_mod_b(grc_blks2.packet_encoder(</u>	
266		samples_per_symbol=2,	
267		bits per symbol=1,	
268		preamble='',	
269		- ccess code='',	
270		pad_for_usrp=True,	
271),	
272		payload_length=0,	
273			
274			
275			
276			
277			
278		# Connections	
279			
280		<pre>self.connect((self.blks2_packet_encoder_0, 0), (self.digital_gmsk_mod_0, 0))</pre>	
281		<pre>self.connect((self.blocks_file_source_0, 0), (self.blocks_throttle_0, 0))</pre>	
282		<pre>self.connect((self.blocks_head_0, 0), (self.blks2_packet_encoder_0, 0))</pre>	
283		<pre>self.connect((self.blocks_multiply_const_vxx_0, 0), (self.uhd_usr_sink_0, 0))</pre>	
284		<pre>self.connect((self.blocks_throttle_0, 0), (self.blocks_head_0, 0))</pre>	
285 286		<pre>self.connect((self.digital_gmsk_mod_0, 0), (self.blocks_multiply_const_vxx_0, 0))</pre>	
287		<pre>def get samp rate(self):</pre>	
288		return self samp, rate	
289		Total out to any _ and	
290		<pre>def set_samp_rate(self, samp_rate):</pre>	
291		self.samp_rate = samp_rate	
292		self.uhd_usrp_sink_0.set_samp_rate(self.samp_rate)	
293		self.blocks throttle 0.set sample rate(self.samp rate)	
294		self.blocks_head_0.set_length(self.samp_rate/4)	
295			
296			
297	cla	ss WRRScheduler():	
298			
299		cw = 0	
300		i = -1	
301		data_set = []	
302		max_s = None	
303		gcd_s = None	
304		len_s = None	
305		counter = ()	
306			
307 308		<pre>definit(self, s = None):</pre>	
308		<pre>selfinit_dataset(s)</pre>	
310		def _init_dataset(self, s):	
311		aerinit_adiaset(seif, s): self.data.set = s	
312		self.udat_set = s self.max_s = max(s, key=lambda x: x[1])[1]	
313		self.god = zgduge(fractions.ged, [weight for data, weight in s])	
314		self len $s = len(s)$	
315			

```
316
            def schedule(self):
                while True:
                    self.i = (self.i + 1) % self.len_s
                   if self.i == 0:
                       self.cw = self.cw - self.gcd_s
                       if self.cw <= 0:</pre>
                          self.cw = self.max_s
                           if self.cw == 0:
324
325
                   if self.data_set[self.i][1] >= self.cw:
                       self._inc_counter(self.data_set[self.i])
                       return self.data_set[self.i]
            def _inc_counter(self, item):
                try:
                    self.counter[item[0]] += 1
                except KeyError:
                    self.counter[item[0]] = 1
334
            def set_data(self, s):
                self.reset()
                self._init_dataset(s)
338
            def reset counter(self);
340
                self.counter = {}
341
            def reset(self):
                self.cw = 0
344
                self.i = -1
                self.data_set = []
                self.max_s = None
self.gcd_s = None
347
                self.len_s = None
                self.reset_counter()
            def get_next(self, n = 1):
                if n > 1:
    return [ self.schedule() for j in range(0,n) ]
                 return self.schedule()
357
        class ack2(gr.top_block, Qt.QWidget):
            def init (self):
                gr.top_block.__init__(self, "Chat2")
Qt.QWidget.__init__(self)
360
361
                self.setWindowTitle("Chat2")
                 qtgui.util.check_set_qss()
364
                try:
                    self.setWindowIcon(Qt.QIcon.fromTheme('gnuradio-grc'))
                except:
367
                   pass
                self.top_scroll_layout = Qt.QVBoxLayout()
                self.setLayout(self.top_scroll_layout)
self.top_scroll = Qt.QScrollArea()
                self.top_scroll.setFrameStyle(Qt.QFrame.NoFrame)
                 self.top_scroll_layout.addWidget(self.top_scroll)
                self.top_scroll.setWidgetResizable(True)
                self.top_widget = Qt.QWidget()
                 self.top_scroll.setWidget(self.top_widget)
                self.top_layout = Qt.QVBoxLayout(self.top_widget)
self.top_grid_layout = Qt.QGridLayout()
                 self.top_layout.addLayout(self.top_grid_layout)
                 self.settings = Qt.QSettings("GNU Radio", "chat2")
381
                 self.restoreGeometry(self.settings.value("geometry").toByteArray())
                 *****
                 *******
                 self.samp_rate = samp_rate = 400000
                 *****
```

390	# Blocks
391	
392	<pre>self.uhd_usrp_source_0 = uhd.usrp_source(</pre>
393	",".join(("", "")),
394	uhd.stream_args(
395	cpu_format="fo32",
396	channels=range(1),
397),
398	
399 100	self.uhd_usrp_source_0.set_samp_rate(samp_rate)
101	self.uhd_usrp_source_0.set_center_freq(1000000000, 0)
102	<pre>self.uhd_usrp_source_0.set_gain(30, 0) self.uhd_usrp_source_0.set_antenna('TX/RX', 0)</pre>
102	self.qtyi_freq_sink x.0 = qtyi.fteq_sink c(
104	1024, faire
105	firdes WIN_BLACKMAN hARRIS, #wintype
106	0, #fo
107	samp_rate, #bw
108	"", #name
109	1 #number of inputs
10)
111	<pre>self.qtgui_freq_sink_x_0.set_update_time(0.10)</pre>
112	<pre>self.qtgui_freq_sink_x_0.set_y_axis(-140, 10)</pre>
113	<pre>self.qtgui_freq_sink_x_0.set_y_label('Relative Gain', 'dB')</pre>
114	<pre>self.qtgui_freq_sink_x_0.set_trigger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "")</pre>
115	self.dtgui freq sink x 0.enable autoscale (False)
116	self.qtgui_freq_sink x_0.enable_grid(False)
117 118	<pre>self.qtgui_freq_sink_x_0.set_fft_average(1.0) self.qtgui_freq_sink_x_0.enable_axis_labels(True)</pre>
110	self.qtyu_freg_sink_v.enable_control_panel(%alse)
120	Sorriggur_reg_bink_x_o.enubre_constor_bunct(rarse)
121	if not True:
122	self.qtqui freq sink x 0.disable legend()
123	
124	<pre>if "complex" == "float" or "complex" == "msg_float":</pre>
125	<pre>self.qtgui_freq_sink_x_0.set_plot_pos_half(not True)</pre>
126	
127	labels = ['', '', '', '', '',
128	
129 🖓 130 🗭	widths = [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1
130 0	colors = ["blue", "red", "green", "black", "cyan",
132	"magenta", "yellow", "dark red", "dark green", "dark blue"]
133 🖯	alphas = [1.0, 1.0, 1.0, 1.0, 1.0,
134 🛱	1.0, 1.0, 1.0, 1.0, 1.0]
135 🖯	<pre>for i in xrange(1):</pre>
136	if len(labels[i]) == 0:
137	<pre>self.qtgui_freq_sink_x_0.set_line_label(i, "Data {0}".format(i))</pre>
138	else:
139	<pre>self.qtgui_freq_sink_x_0.set_line_label(i, labels[i])</pre>
140	self.qtgui_freq_sink_x_0.set_line_width(i, widths[i])
141	self.qtgui_freq_sink_x_0.set_line_color(i, colors[i])
142 - 143	<pre>self.qtgui_freq_sink_x_0.set_line_alpha(i, alphas[i])</pre>
145	<pre>selfgtgui_freq_sink_x_0_win = sip.wrapinstance(self.qtgui_freq_sink_x_0.pyqwidget(), Qt.QWidget)</pre>
145	self. <u>yyyy</u> ire <u>l</u> sinc <u>a</u> _widet(self.yyy) self.top grid layout.addWidet(self.yyyi qrui freq sink x 0 win)
146	<pre>self.low pass filter 0 = filter.fir filter ccf(1, firdes.low pass(</pre>
147	1, samp rate, 200000, 50000, firdes.WIN HAMMING, 6.76))
148	self.digital gmsk_demod_1 = digital.gmsk_demod(
149	<pre>samples per_symbol=2,</pre>
150	gain_mu=0.175,
151	mu=0.5,
152	<pre>omega_relative_limit=0.005,</pre>
153	freq_error=0.0,
154	verbose-False,
155 156	log=False,
156 157) self.blocks_multiply_const_vxx_1 = blocks.multiply_const_vcc((1,))
157	<pre>self.blocks_multply_const_vxx_i = blocks.imultply_const_vcc((i,)) self.blocks_file sink (i = blocks.file sink(i:sizeof char*1, '/home/b/chat test2.txt', False)</pre>
150	self.blocks_file_sink_iset_unbuffered(Palse)_sizeD_unal_i, /nome/b/unal_estz.ckt, false)
160	self.blks2 packet_decoder [= grc_blks2.packet_demod_b(grc_blks2.packet_decoder(
161	access code='',
162	threshold=-1,
163	callback=lambda ok, payload: self.blks2 packet decoder 1.recv pkt(ok, payload),

<pre>idea), }, } idea), idea), idea), idea ; idea ; idea </pre>	
<pre>67 67 68 69 69 69 69 69 60 60 60 60 60 60 60 60 60 60 60 60 60</pre>	
<pre>68 69 69 69 69 69 69 69 69 60 60 60 60 60 60 60 60 60 60 60 60 60</pre>	
<pre> ####################################</pre>	
<pre>f Connections ff Connect(self.blks2_packet_decoder_1, 0), (self.blocks_file_sink_1, 0)) self.connect((self.digital_gmsk_demod_1, 0), (self.blks2_packet_decoder_1, 0)) self.connect((self.digital_gmsk_demod_1, 0), (self.blks2_packet_decoder_1, 0)) self.connect((self.low_pass_filter_0, 0), (self.digital_gmsk_demod_1, 0)) self.connect((self.set_samp_source_0, 0), (self.digital_gmsk_demod_1, 0)) self.connect((self.set_samp_source_0, 0), (self.digital_gmsk_demod_1, 0)) self.settings = Qt.QSettings("GNV Radio", "chat2") self.settings = Qt.QSettings("GNV Radio", "chat2") self.settings.setValue("geometry", self.saveGeometry()) event.accept() def get_samp_rate(self): return self.samp_rate self.udu_stp_source(.set_samp_rate(self.samp_rate) self.dud_ustp_source(.set_samp_rate(self.samp_rate) self.low_pass_filter_0.set_taps(firdes.low_pass(1, self.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76)) self.low_pass_filter_0.set_taps(firdes.low_pass(1, self.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76)) </pre>	
<pre>file file file file file file file file</pre>	
<pre>self.connect((self.blks2_packet_decoder_1, 0), (self.blocks_file_sink_1, 0)) self.connect((self.blocks_multiply_const_vxx_1, 0), (self.low_pass_filter_0, 0)) self.connect((self.low_pass_filter_0, 0), (self.digital_gmsk_demod_1, 0)) self.connect((self.low_pass_filter_0, 0), (self.digital_gmsk_demod_1, 0)) self.connect((self.uhd_usrp_source_0, 0), (self.blocks_multiply_const_vxx_1, 0)) def closeEvent(self, event): self.settings = Ot.OSettings("GNV Radio", "chat2") self.settings.setValue("geometry", self.saveGeometry()) event.accept() def get_samp_rate(self): return self.samp_rate def set_samp_rate(self, samp_rate): self.uhd_usrp_source_0.set_samp_rate(self.samp_rate) self.uhd_usrp_source(0, set_frequency_range(0, self.samp_rate) self.ubd_usrp_source_0.set_taps(firdes.low_pass(1, self.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76)) </pre>	
<pre>self.connect((self.block_multiply_const_vxx_1, 0), (self.low_pass_filter_0, 0)) self.connect((self.digital_gmsk_demod_1, 0), (self.blks2_packet_decoder_1, 0)) self.connect((self.low_pass_filter_0, 0), (self.digital_gmsk_demod_1, 0)) self.connect((self.low_pass_filter_0, 0), (self.digital_gmsk_demod_1, 0)) self.connect((self.uhd_usrp_source_0, 0), (self.blocks_multiply_const_vxx_1, 0)) def closeEvent(self, event): self.settings = Qt.QSettings("GNU Radio", "chat2") self.settings.setValue("geometry", self.saveGeometry()) event.accept() def get_samp_rate(self): return self.samp_rate def set_samp_rate(self, samp_rate) self.qtgui_freq_sink_x_0.set_frequency_range(0, self.samp_rate) self.qtgui_freq_sink_x_0.set_frequency_range(0, self.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76)) </pre>	
<pre>74 self.connect((self.digital_gmsk_demod_1, 0), (self.blks2_packet_decoder_1, 0)) 75 self.connect((self.low_pass_filter_0, 0), (self.digital_gmsk_demod_1, 0)) 76 self.connect((self.low_pass_filter_0, 0), (self.digital_gmsk_demod_1, 0)) 77 self.connect((self.low_pass_filter_0, 0), (self.digital_gmsk_demod_1, 0)) 78 self.connect((self.low_pass_filter_0, 0), (self.digital_gmsk_demod_1, 0)) 79 self.connect((self.uhd_usrp_source_0, 0), (self.blocks_multiply_const_vxx_1, 0)) 79 def closeEvent(self, event): 79 self.settings = Qt.QSettings("GNV Radio", "chat2") 79 self.settings.setValue("geometry", self.saveGeometry()) 82 event.accept() 83 def get_samp_rate(self): 79 return self.samp_rate 86 def set_samp_rate(self, samp_rate) 87 self.gamp_rate = samp_rate 88 self.uhd_usrp_source_0.set_samp_rate(self.samp_rate) 89 self.qtpui_freq_sink_x_0.set_frequency_range(0, self.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76)) 89</pre>	
<pre>self.connect((self.low_pass_filter_0, 0), (self.digital_gmsk_demod_1, 0)) self.connect((self.low_pass_filter_0, 0), (self.digital_gmsk_demod_1, 0)) self.connect((self.uhd_usr_gource_0, 0), (self.digital_gmsk_demod_1, 0)) self.connect((self.samp_rate): self.samp_rate(self, samp_rate(self.samp_rate) self.uhd_usr_gource_0.set_samp_rate(self.samp_rate) self.low_pass_filter_0.set_taps(firdes.low_pass(1, self.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76)) </pre>	
<pre>self.connect((self.low_pass_filter_0, 0), (self.qtgui_freq_sink_x_0, 0)) self.connect((self.uhd_usrp_source_0, 0), (self.blocks_multiply_const_vxx_1, 0)) def closeEvent(self, event): self.settings = Qt_QSettings("GNU Radio", "chat2") self.settings.setValue("geometry", self.saveGeometry()) event.accept() def get_samp_rate(self): return self.samp_rate def set_samp_rate(self, samp_rate): self.sgamp_rate(self, samp_rate) self.uhd_usrp_source_0.set_samp_rate(self.samp_rate) self.qtgui_freq_sink_x_0.set_frequency_range(0, self.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76)) </pre>	
<pre>def closeEvent(self, event): self.settings = Qt.QSettings("GNU Radio", "chat2") self.settings.setValue("geometry", self.saveGeometry()) event.accept() def get_samp_rate(self): return self.samp_rate def set_samp_rate(self, samp_rate): self.setings_source_0.set_samp_rate(self.samp_rate) self.qtgui_freq_sink_x_0.set_frequency_range(0, self.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76)) self.low_pass_filter_0.set_taps(firdes.low_pass(1, self.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76)) </pre>	
<pre>def closeEvent(self, event): self.settings = Qt.QSettings("GNU Radio", "chat2") self.settings.setValue("geometry", self.saveGeometry()) event.accept() def get_samp_rate(self): return self.samp_rate def set_samp_rate(self): self.samp_rate(self, samp_rate): self.samp_rate(self, samp_rate): self.gamp_rate(self, samp_rate) self.uhd_usrp_source_0.set_samp_rate(self.samp_rate) self.qtgui_freq_sink_x_0.set_frequency_range(0, self.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76)) </pre>	
<pre>solf.settings = Qt.QSettings("GNU Radio", "chat2") solf.settings.setValue("geometry", solf.saveGeometry()) event.accept() def get_samp_rate(solf): return solf.samp_rate def set_samp_rate(solf, samp_rate): solf.samp_rate(solf, samp_rate) solf.uhd_usrp_source_0.set_samp_rate(solf.samp_rate) solf.qtgui_freq_sink_x_0.set_frequency_range(0, solf.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76)) solf.low_pass_filter_0.set_taps(firdes.low_pass(1, solf.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76))</pre>	
<pre>self.settings.setValue("geometry", self.saveGeometry()) sevent.accept() def get_samp_rate(self): return self.samp_rate def set_samp_rate(self, samp_rate): self.samp_rate(self, samp_rate): self.samp_rate = samp_rate self.uhd_usrp_source_0.set_samp_rate(self.samp_rate) self.uhd_usrp_source_0.set_frequency_range(0, self.samp_rate) self.low_pass_filter_0.set_taps(firdes.low_pass(1, self.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76)) </pre>	
<pre>event.accept() def get_samp_rate(self): return self.samp_rate def set_samp_rate(self, samp_rate): self.gamp_rate(self, samp_rate): self.gamp_rate = samp_rate self.uhd_usrp_source_0.set_samp_rate(self.samp_rate) self.qtgui_freq_sink_x_0.set_frequency_range(0, self.samp_rate) self.low_pass_filter_0.set_taps(firdes.low_pass(1, self.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76)) </pre>	
<pre>def get_samp_rate(self): return self.samp_rate def set_samp_rate(self, samp_rate): self.gamp_rate(self, samp_rate): self.gamp_rate(self, samp_rate) self.uhd_usrp_source_0.set_samp_rate(self.samp_rate) self.qtgui_freq_sink_x_0.set_frequency_range(0, self.samp_rate) self.low_pass_filter_0.set_taps(firdes.low_pass(1, self.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76)) </pre>	
<pre>def get_samp_rate(self): return self.samp_rate def set_samp_rate(self, samp_rate): self.samp_rate(self, samp_rate): self.uhd_usrp_source_0.set_samp_rate(self.samp_rate) self.uhd_usrp_source_0.set_frequency_range(0, self.samp_rate) self.low_pass_filter_0.set_taps(firdes.low_pass(1, self.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76)) </pre>	
<pre>85 return self.samp_rate 86 87 def set_samp_rate(self, samp_rate): 88 self.samp_rate = samp_rate 89 self.uhd_usrp_source_0.set_samp_rate(self.samp_rate) 90 self.qtqui_freq_sink_x_0.set_frequency_range(0, self.samp_rate) 91 self.low_pass_filter_0.set_taps(firdes.low_pass(1, self.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76)) 93</pre>	
<pre>87 def set_samp_rate(self, samp_rate): 88 self.samp_rate = samp_rate 99 self.uhd_usrp_source_0.set_samp_rate(self.samp_rate) 90 self.qtgui_freq_sink_x_0.set_frequency_range(0, self.samp_rate) 91 self.low_pass_filter_0.set_taps(firdes.low_pass(1, self.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76)) 92 93</pre>	
<pre>88 self.gamp_rate = samp_rate 89 self.uhd_usrp_source_0.set_samp_rate(self.samp_rate) 90 self.qtgui_freq_sink_x_0.set_frequency_range(0, self.samp_rate) 91 self.low_pass_filter_0.set_taps(firdes.low_pass(1, self.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76)) 93</pre>	
<pre>89 self.uhd_usrp_source_0.set_samp_rate(self.samp_rate) 90 self.qtgui_freq_sink_x_0.set_frequency_range(0, self.samp_rate) 91 self.low_pass_filter_0.set_taps(firdes.low_pass(1, self.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76)) 92 93</pre>	
90 self.qtgui_freq_sink_x_0.set_frequency_range(0, self.samp_rate) 91 self.low_pass_filter_0.set_taps(firdes.low_pass(1, self.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76)) 93 93	
91 91 self.low_pass_filter_0.set_taps(firdes.low_pass(1, self.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76)) 92 93	
92 93	
93	
95	
96 from distutils.version import StrictVersion	
97 dif StrictVersion(Qt.qVersion()) >= StrictVersion("4.5.0"):	
<pre>98 style = gr.prefs().get_string('qtgui', 'style', 'raster')</pre>	
99 C Qt.QApplication.setGraphicsSystem(style) 00 gapp = Qt.QApplication(sys.argv)	
000 gapp = Qt.QApplication(sys.argv) 301	
data_csv = open('/home/b/20161005 140846.csv', 'r')	
303 <u>csyreader</u> = csy.reader(data_csy)	
header = next(csvreader)	
<pre>data_rows = []</pre>	
006 for row in csvreader:	
007 data_rows.append(row)	
data_csv.close()	
10 for x in range(0,14):	
out = open('file' + str(x+1) + '.txt', 'w')	
for y in range (0, 100):	
<pre>ii3 out.write(data_rows[y][x+6]+' '+str(x+1)+'0'+str(y+1)+'\n')</pre>	
14 O out.close()	
115 the shat?()	
16 tb = chat2() 17 tb.start()	
1 Unitary (1) Unit	
19 while True:	
<pre>20 f = open('/home/b/chat_test1.txt', 'r')</pre>	
f.flush()	
22 f.readline()	
<pre>check = f.readline()</pre>	
if (check != ""): 25 print_"START"	
226 time.sleep(1)	
27 O break	
128 tb.stop()	
29	
<pre>weight_file = open('/home/b/chat_test1.txt', 'r')</pre>	
<pre>weight_file.flush()</pre>	
<pre>32 weight_file.readline() wight_file.readline()</pre>	
<pre>weights = weight_file.readline() w_list = weights.split(",")</pre>	
<pre>Main Main Main Main Main Main Main Main</pre>	
<pre>form in range(0, len(w_list)):</pre>	
337 w_list[m] = int(w_list[m])	

538		
539	sensors = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N']	
540		
541	global i, al, bl, cl, dl, el, fl, gl, hl, il, jl, kl, ll, ml, nl	
542 543	i = 0 a1 = 0	
544		
545		
546	d1 = 0	
547	e1 = 0	
548	f1 = 0	
549	g1 = 0 h1 = 0	
550 551		
552		
553	k1 = 0	
554	11 = 0	
555	m1 = 0	
556	nl = 0	
557	global data0, data1, data2, data3, data4, data5, data6, data7, data8, data9, data10, data11, data12, data13	
559	groon acces, accer, accer, acces, acces, acces, acces, acces, acces, accers, accer	
560	<pre>read1 = open('/home/b/file1.txt', 'r')</pre>	
561	data0 = read1.readlines()	
562	<pre>length1 = len(data0)</pre>	
563	readl.close()	
564 565	<pre>read2 = open('/home/b/file2.txt', 'r')</pre>	
566	data = read2.readlines()	
567	length2 = len(data1)	
568	read2.close()	
569		
570 571	<pre>read3 = open('/home/b/file3.txt', 'r') data2 = read3.readlines()</pre>	
572	langth = len(dta2)	
573	read3.close()	
574		
575	<pre>read4 = open('/home/b/file4.txt', 'r') data3 = read4.readlines()</pre>	
576 577	data3 = read4.readlines() length4 = len(data3)	
578	read4.close()	
579		
580	<pre>read5 = open('/home/b/file5.txt', 'r')</pre>	
581 582	<pre>data4 = read5.readlines() length5 = len(data4)</pre>	
583	length - len(latat) read5.close()	
584		
585	<pre>read6 = open('/home/b/file6.txt', 'r')</pre>	
586	data5 = read6.readlines()	
587	<pre>length6 = len(data5)</pre>	
588 589	read6.close()	
590	<pre>read7 = open('/home/b/file7.txt', 'r')</pre>	
591	<pre>data6 = read7.readlines()</pre>	
592	length7 = len(data6)	
593	read7.close()	
594 595	xxx49 = xxxx/1/home/h/file8 + x+1 + 1+1	
595	<pre>read8 = open('/home/b/file8.txt', 'r') data7 = read8.readlines()</pre>	
597	length8 = len(data7)	
598	read8.close()	
599		
600	<pre>read9 = open('home/b/file9.txt', 'r') dte0 = read0 = read</pre>	
601 602	<pre>data8 = read9.readlines() length9 = len(data8)</pre>	
603	reads-lose()	
604		
605	<pre>read10 = open('/home/b/file10.txt', 'r')</pre>	
606	<pre>data9 = read10.read1ines()</pre>	
607 608	length10 = len(data9) read10.close()	
609		
610	<pre>read11 = open('/home/b/file11.txt', 'r')</pre>	
611	data10 = read11.readlines()	

612	<pre>length11 = len(data10)</pre>
613	readl.close()
614	
615	<pre>read12 = open('/home/b/file12.txt', 'r')</pre>
616	datal1 = readl2.readlines()
617	<pre>length12 = len(data11)</pre>
618	read12.close()
619	
620	<pre>read13 = open('/home/b/file13.txt', 'r')</pre>
621	data12 = read13.readlines()
622	length13 = len(data12)
623	read13.close()
624	
625	<pre>read14 = open('/home/b/file14.txt', 'r')</pre>
626	<pre>data13 = read14.readlines()</pre>
627	<pre>length14 = len(data13)</pre>
628	read14.close()
629	
630	<pre>data = list(zip(sensors, w_list))</pre>
631	sched = WRRScheduler(data)
632	
633 634	<pre>global a_tot, b_tot, c_tot, d_tot, e_tot, f_tot, g_tot, h_tot, i_tot, j_tot, k_tot, l_tot, m_tot, n_tot, switch a_tot = 0</pre>
635	a_000 - 0 b_tot = 0
636	cot = 0
637	d tot = 0
638	 e tot = 0
639	$f_{tot} = 0$
640	- tot = 0
641	$h_{tot} = 0$
642	i_tot = 0
643	j_tot = 0
644	k_tot = 0
645	l_tot = 0
646	m_tot = 0
647	n_tot = 0
648 649	x = 0
650	result = []
651 👳	while $(x < 150)$:
652	<pre>choose = sched.get_next()</pre>
653 Ϋ	if choose[0] == 'A':
654	a_tot += 1
655 🖯	<pre>if (a_tot <= length1):</pre>
656	result.append(choose[0])
657 D	
659	<pre>elif choose[0] == 'B'; bet f= '</pre>
660 🖯	<pre>b_tot += 1 if (b_tot <= length2):</pre>
661	
662	result.append(choose[0])
	result.append(choose[0]) x += 1
663	<pre>result.append(choose[0]) x += 1 elif choose[0] == 'C':</pre>
	x += 1
663 🖯	<pre>x += 1 elif choose[0] == 'C':</pre>
663 🖯	<pre>x += 1 elif choose[0] == 'C': c_tot += 1</pre>
663 🛡 664 665 🤤	<pre>x += 1 elif choose[0] == 'C': c_tot += 1 if (c_tot <= length3):</pre>
663 📮 664 665 📮 666	<pre>x += 1 elif choose[0] == 'C':</pre>
663 - 664 - 665 - 666 - 668 - 669 -	<pre>x += 1 elif choose[0] == 'c'; c_tot += 1 if (c_tot <= length3): result.append(choose[0]) x += 1 elif choose[0] == 'D'; d_tot += 1</pre>
663 - 664 - 665 - 666 - 668 - 669 - 670 -	<pre>x += 1 elif choose[0] == 'C': c_tot += 1 if (c_tot <= length3): result.append(choose[0]) x += 1 elif choose[0] == 'D': d_tot += 1 if (d_tot <= length4):</pre>
663 664 665 666 667 668 669 670 671	<pre>x += 1 elif choose[0] == 'C':</pre>
663 664 665 666 667 668 669 670 671 672 672 672 672 672 673 674 675 737 747	<pre>x += 1 elif choose[0] == 'c'; c_tot += 1 if (c_tot <= length3): result.append(choose[0]) x += 1 elif choose[0] == 'p'; d_tot += 1 if (d_tot <= length4): result.append(choose[0]) x += 1</pre>
663 664 665 666 667 668 669 670 671 672 673 0	<pre>x += 1 elif choose[0] == 'D': c_tot += 1 if (c_tot <= length3): result.append(choose[0]) x += 1 elif choose[0] == 'D': d_tot += 1 if (d_tot <= length4): result.append(choose[0]) x += 1 elif choose[0] == 'B':</pre>
663 664 665 667 668 669 671 672 673 674	<pre>x += 1 elif choose[0] == 'C': c_tot += 1 if (c_tot <= length3): result.append(choose[0]) x += 1 elif choose[0] == 'D': d_tot += 1 if (d_tot <= length4): result.append(choose[0]) x += 1 elif choose[0] == 'E': e_tot += 1</pre>
663 I 664 I 665 I 666 I 667 I 668 I 670 I 671 I 673 I 674 I 675 I	<pre>x += 1 elif choose[0] == 'C': c_tot += 1 if (c_tot <= length3): result.append(choose[0]) x += 1 elif choose[0] == 'D': d_tot += 1 if (d_tot <= length4): result.append(choose[0]) x += 1 elif choose[0] == 'B': e_tot += 1 if (e_tot <= length5):</pre>
663 0 664 666 666 0 6667 0 668 0 670 0 671 671 0 673 0 674 675 0 675 0	<pre>x += 1 elif choose[0] == '0': c_tot += 1 if (c_tot <= length3): result.append(choose[0]) x += 1 elif choose[0] == 'D': d_tot += 1 if (d_tot <= length4): result.append(choose[0]) x += 1 elif choose[0] == 'E': e_tot += 1 if (e_tot <= length5): result.append(choose[0])</pre>
663 I 664 I 665 I 666 I 667 I 668 I 670 I 671 I 673 I 674 I 675 I	<pre>x += 1 elif choose[0] == 'C': c_tot += 1 if (c_tot <= length3): result.append(choose[0]) x += 1 elif choose[0] == 'D': d_tot += 1 if (d_tot <= length4): result.append(choose[0]) x += 1 elif choose[0] == 'B': e_tot += 1 if (e_tot <= length5): result.append(choose[0]) x += 1</pre>
663 0 664 0 665 0 666 0 668 0 670 0 671 0 672 0 673 0 674 0 675 0 676 0 677 0	<pre>x += 1 elif choose[0] == '0': c_tot += 1 if (c_tot <= length3): result.append(choose[0]) x += 1 elif choose[0] == 'D': d_tot += 1 if (d_tot <= length4): result.append(choose[0]) x += 1 elif choose[0] == 'E': e_tot += 1 if (e_tot <= length5): result.append(choose[0])</pre>
663 0 664 0 665 0 666 0 668 0 669 0 671 0 672 0 673 0 674 0 675 0 676 0 677 0 677 0 677 0 677 0	<pre>x += 1 elif choose[0] == 'C': c_tot += 1 if (c_tot <= length3): result.append(choose[0]) x += 1 elif choose[0] == 'D': d_tot += 1 if (d_tot <= length4): result.append(choose[0]) x += 1 elif choose[0] == 'B': e_tot += 1 if (e_tot <= length5): result.append(choose[0]) x += 1 elif choose[0] == 'P':</pre>
663 0 664 0 665 0 666 0 667 0 668 0 671 0 673 0 674 0 675 0 676 0 677 0 676 0 677 0 678 0 679 0	<pre>x += 1 elif chose[0] == 'c': c_tot += 1 if (c_tot <= length3): result.append(chose[0]) x += 1 elif chose[0] == 'p': d_tot += 1 if (d_tot <= length4): result.append(chose[0]) x += 1 elif chose[0] == 'g': e_tot += 1 if (e_tot <= length5): result.append(chose[0]) x += 1 elif chose[0] == 'p': f_tot <= 1</pre>
663 0 664 0 665 0 666 0 667 0 671 0 671 0 671 0 671 0 671 0 671 0 677 0 677 0 677 0 677 0 677 0 677 0 677 0 679 0 680 0	<pre>x += 1 elif choose[0] == 'C': c_tot += 1 if (c_tot <= length3): result.append(choose[0]) x += 1 elif choose[0] == 'D': d_tot += 1 if (d_tot <= length4): result.append(choose[0]) x += 1 elif choose[0] == 'E': e_tot += 1 if (e_tot <= length5): result.append(choose[0]) x += 1 elif choose[0] == 'F': f_tot += 1 if (f_tot <= length6): result.append(choose[0]) x += 1</pre>
663 0 664 0 665 0 666 0 667 0 670 0 671 0 673 0 674 0 675 0 676 0 677 0 678 0 679 0 680 0 681 0 682 0 683 0	<pre>x += 1 elif choose[0] == '0'; c_tot += 1 if (c_tot <= length3): result.append(choose[0]) x += 1 elif choose[0] == 'D'; d_tot += 1 if (d_tot <= length4): result.append(choose[0]) x += 1 elif choose[0] == 'E': e_tot += 1 if (e_tot <= length5): result.append(choose[0]) x += 1 elif choose[0] == 'F': f_tot += 1 if (f_tot <= length6): result.append(choose[0]) x += 1 elif choose[0] == 'C':</pre>
663 0 664 0 665 0 666 0 667 0 670 0 671 0 672 0 673 0 674 0 675 0 676 0 677 0 678 0 679 680 682 0 682 0 682 0 682 0 683 0 684 0	<pre>x += 1 elif chose[0] == 'C': c_tot += 1 if (c_tot << length3): result.append(chose[0]) x += 1 elif chose[0] == 'D': d_tot += 1 if (d_tot << length4): result.append(chose[0]) x += 1 elif chose[0] == 'B': e_tot += 1 if (e_tot << length5): result.append(chose[0]) x += 1 elif chose[0] == 'P': f_tot += 1 if (f_tot <= length6): result.append(chose[0]) x += 1 elif chose[0] == 'G': g_tot += 1</pre>
663 0 664 0 665 0 666 0 667 0 670 0 671 0 672 0 673 0 674 0 675 0 676 0 677 0 678 0 679 0 680 0 681 0 682 0	<pre>x += 1 elif choose[0] == '0'; c_tot += 1 if (c_tot <= length3): result.append(choose[0]) x += 1 elif choose[0] == 'D'; d_tot += 1 if (d_tot <= length4): result.append(choose[0]) x += 1 elif choose[0] == 'E': e_tot += 1 if (e_tot <= length5): result.append(choose[0]) x += 1 elif choose[0] == 'F': f_tot += 1 if (f_tot <= length6): result.append(choose[0]) x += 1 elif choose[0] == 'C':</pre>

586	result.append(choose[0])
87	$x \neq z$
88	elif choose[0] == 'H':
89	$h \cot t = 1$
90	<pre>if (h_tot <= length8):</pre>
91	result.append(choose[0])
92	$\mathbf{x} + \mathbf{z} = 1$
93	<pre>elif choose[0] == 'I':</pre>
94	$i_{tot} \neq 1$
95	<pre>if (i_tot <= length9):</pre>
96	result.append(choose[0])
97	x += 1
98	elif choose[0] == 'J':
99	j_tot += 1
00	<pre>if (j_tot <= length10):</pre>
01	result.append(choose[0])
02	x += 1
03	<pre>elif choose[0] == 'K':</pre>
04	k_{\pm} tot += 1
05	<pre>if (k_tot <= length1):</pre>
06	result.append(choose[0])
07	$\mathbf{x} + \mathbf{z} = 1$
08	elif choose[0] == 'L':
09	$\frac{1}{16} \cot \theta = 1$
10 11	if (l_tot <= length12): result = negred(choces(0))
	result.append(choose[0])
12 13	$x \neq 1$
13	elif choose[0] == 'M':
15	<pre>m_tot += 1 if (m_tot <= length13):</pre>
16	result.append(choose[0])
17	x += 1 x
18	$\operatorname{elif}_{chose}[0] = 'N';$
19	$n to t \neq 1$
20	if (n_tot <= length14):
21	result.append(choose[0])
22	x += 1
723	
724	while i < (len(result)):
725	if result[i] == 'A':
726	switch = 'a'
727	<pre>tb = text_tx()</pre>
728	tb.start()
729	time.sleep(1)
730	tb.stop()
731	i += 1
732	al += 1
733	<pre>elif result[i] == 'B':</pre>
734	switch = 'b'
735	tb = text_tx()
736	tb.start()
737	time.sleep(1)
738	tb.stop()
739	i += 1
740	b1 += 1
741	<pre>elif result[i] == 'C':</pre>
742	switch = 'c'
743	$tb = text_tx()$
744	tb.start()
745	time.sleep(1)
746	tb.stop()
747	i += 1
748 749	
149	<pre>elif result[i] == 'D':</pre>
750	switch = 'd'
750 751	tb = text_tx()
750 751 752	<pre>tb = text_tx() tb.start()</pre>
750 751 752 753	<pre>tb = text_tx() tb.start() time.sleep(1)</pre>
750 751 752 753 754	<pre>tb = text_tx() tb.start() time.sleep(1) tb.stop()</pre>
750 751 752 753 754 755	<pre>tb = text_tx() tb.start() time.sleep(1) tb.stop() i += 1</pre>
750 751 752 753 754 755 756	<pre>tb = text_tx() tb.start() time.sleep(1) tb.stop() i += 1 d1 += 1</pre>
750 751 752 753 754 755 756 757	<pre>tb = text_tx() tb.start() time.sleep(1) tb.stop() i += 1 d1 += 1 elif result[i] == 'E':</pre>
750 751 752 753 754 755 756	<pre>tb = text_tx() tb.start() time.sleep(l) tb.stop() i += 1 dl += 1 elif result[i] == 'E': switch = 'e'</pre>

/Τ	1	C	•	`
(Image	continued	trom	previous	nage)
(IIIIage	commaca		p10,10,40	page/

760	tb.start()
761	time.sleep(1)
762	tb.stop()
763 764	i += 1 e1 += 1
765	<pre>elif result[i] == 'F':</pre>
766	switch = 'f'
767	<pre>tb = text_tx()</pre>
768	tb.start()
769	time.sleep(1)
770	tb.stop()
771 772	i += 1 f1 += 1
772	<pre>f1 += 1 elif result[i] == 'G':</pre>
774	switch = 'g'
775	tb = text_tx()
776	tb.start()
777	time.sleep(1)
778	tb.stop()
779	i += 1
780 781	g1 += 1 elif result[i] == 'H':
781	switch = 'h'
783	tb = text_tx()
784	tb.start()
785	time.sleep(1)
786	tb.stop()
787	i += 1
788	$h1 \neq 1$
789	<pre>elif result[i] == 'I': switch = 'i'</pre>
790 791	tb = text_tx()
791	tb.start()
793	time.sleep(1)
794	tb.stop()
795	i += 1
796	i1 += 1
797	elif result[i] == 'J':
798 799	switch = 'j'
799 800	<pre>tb = text_tx() tb.start()</pre>
801	time.sleep(1)
802	tb.stop()
803	i += 1
804	j1 += 1
805	<pre>elif result[i] == 'K':</pre>
806	switch = 'k'
807 808	<pre>tb = text_tx() tb.start()</pre>
808	tb.start() time.sleep(1)
810	tb.stop()
811	i += 1
812	k1 += 1
813	<pre>elif result[i] == 'L':</pre>
814	switch = 'l'
815	tb = text_tx()
816	tb.start()
817 818	time.sleep(1) tb.stop()
819	i += 1
820	11 += 1
821	<pre>elif result[i] == 'M':</pre>
822	switch = 'm'
823	<pre>tb = text_tx()</pre>
824	tb.start()
825	time.sleep(1)
826	tb.stop() i += 1
	1 += 1 m1 += 1
827 828	
827 828 829	<pre>elif result[i] == 'N':</pre>
828	<pre>elif result[i] == 'N': switch = 'n'</pre>
828 829 830 831	<pre>switch = 'n' tb = text_tx()</pre>
828 829 830	switch = 'n'

834		tb.stop()
835		i += 1
836		n1 += 1
837		else:
838		print "Error: Empty Result"
839		tb = ack2()
840		tb.start()
841		while True:
842		<pre>ack_file = open('/home/b/chat_test2.txt', 'r')</pre>
843		ack_file.flush()
844		ack_file.readline()
845		<pre>ack0 = ack_file.readline()</pre>
846		if $(ack0 == 'ACK \):$
847		print, "RECEIVING ACK"
848		time.sleep(1)
849		break
850		ack_file.close()
851		time.sleep(1)
852		tb.stop()
853		
854		switch = 'eof'
855		tb = text_tx()
856		tb.start()
857		time.sleep(1)
858		tb.stop()
859		
860		print('done')
861		exit()
862		
863		<pre>def quitting():</pre>
864		tb.stop()
865		tb.wait()
866		<pre>qapp.connect(qapp, Qt.SIGNAL("aboutToQuit()"), quitting)</pre>
867		<pre>qapp.exec_()</pre>
868		
869		
870	if	name == 'main':
871		main()

Figure A.2: Weighted Round-Robin Algorithm for Field Sensor Radio

1		sr/bin/env python2
2		- adding: utf-8 -*-
3		
4		U Radio Python Flow Graph tle: Chat
6		nerated: Thu Feb 10 09:30:28 2022
7		
8		
9 🕨	0 if	name_ == ' main ':
10		import ctypes
11		import sys
12		if sys.platform.startswith('linux'):
13		try:
14		<pre>x11 = ctypes.cdll.LoadLibrary('libX11.so')</pre>
15		x11.XInitThreads()
16 17		except: print "Warning: failed to XInitThreads()"
18		print, "warning: failed to Anicinfeads()"
19	impo	rt
37	1	
38		
39	clas	s chat(gr.top_block, Qt.QWidget):
40		
41		definit(self):
42		gr.top_blockinit(self, "Chat")
43		Qt_QWidgetinit(self)
44 45		<pre>self.setWindowTitle("Chat") qtgui.util.check set qss()</pre>
46		dtgui.utii.cnetx_set_qss() try:
47		self.setWindowIcon(Qt.QIcon.fromTheme('qnuradio-gro'))
48		except:
49		pass
50		<pre>self.top_scroll_layout = Qt.QVBoxLayout()</pre>
51		self.setLayout(self.top_scroll_layout)
52		<pre>self.top_scroll = Qt.QScrollArea()</pre>
53		self.top_scroll.setFrameStyle(Qt.QFrame.NoFrame)
54 55		self.top_scroll_layout.addWidget(self.top_scroll)
56		<pre>self.top_scroll.setWidgetResizable(True) self.top_widget = Qt.QWidget()</pre>
57		self.top_scroll.setWidget(self.top_widget)
58		self.top_layout = Qt.QVBoxLayout(self.top_widget)
59		<pre>self.top_grid_layout = Qt.QGridLayout()</pre>
60		self.top_layout.addLayout(self.top_grid_layout)
61		
62		self.settings = Qt.QSettings("GNV Radio", "chat")
63 64		<pre>self.restoreGeometry(self.settings.value("geometry").toByteArray())</pre>
65		
66		
67		# Variables
68		
69		self.samp_rate = samp_rate = 400000
70		
71		
72		# Blocks
73 74		self.uhd usrp sink 1 = uhd.usrp sink(
75		set.tund_usep_stnk_i - und.usep_stnk(",^.join(("", ")),
76		uhd.stream_args(
77		cpu format="fc32",
78		channels=range(1),
79),
80)
81		<pre>self.uhd_usrp_sink_1.set_samp_rate(samp_rate)</pre>
82		<pre>self.uhd_usrp_sink_1.set_time_now(uhd.time_spec(time.time()), uhd.ALL_MBOARDS)</pre>
83		self.uhd_usrp_sink_l.set_center_freq(100000000, 0)
84 85		self uhd usrp sink 1.set gain(40, 0)
85		<pre>self.uhd_usrp_sink_1.set_antenna('RX2', 0) self.digital_gmsk_mod_0 = digital.gmsk_mod(</pre>
87		samlespersymbol=2.
88		bt=0.35,
89		verbose=False,
90		log=False,
91		X

92			<pre>self.blocks throttle 0 = blocks.throttle(gr.sizeof char*1, samp rate,True)</pre>	
93			<pre>self.blocks_multiply_const_vxx 1 = blocks.multiply_const_vcc((1,))</pre>	
94			<pre>self.blocks head 0 = blocks.head(gr.sizeof char*1, 100000)</pre>	
95			self.blocks file source 0 = blocks.file source(gr.sizeof char*1, '/home/c/chat3', True)	
96			self.blocks_file_source_0.set_begin_tag(pmt.PMT_NIL)	
97			self.blks2_packet_encoder_0 = grc_blks2.packet_mod_b(grc_blks2.packet_encoder(
98			<pre>samples_per_symbol=2,</pre>	
99			<pre>bits_per_symbol=1,</pre>	
100			preamble='',	
101			access_code='',	
102			pad_for_usrp=True,	
103),	
104			payload_length=0,	
105)	
106				
107				
108				
109				
110			# Connections	
111				
112			<pre>self.connect((self.blks2_packet_encoder_0, 0), (self.digital_gmsk_mod_0, 0))</pre>	
113			<pre>self.connect((self.blocks_file_source_0, 0), (self.blocks_throttle_0, 0))</pre>	
114			<pre>self.connect((self.blocks_head 0, 0), (self.blk2_packet_encoder_0, 0))</pre>	
115			self.connect((self.blocks_multiply_const_vxx_1, 0), (self.uhd_usrp_sink_1, 0))	
116 117			self.connect((self.blocks_throttle_0, 0), (self.blocks_head_0, 0))	
			<pre>self.connect((self.digital_gmsk_mod_0, 0), (self.blocks_multiply_const_vxx_1, 0))</pre>	
118 119		daf	interpret/solf_enerty.	
120		der	<pre>closeEvent(self, event): self.settings = Qt.QSettings("GNU Radio", "chat")</pre>	
120			seri.settings.etVale("geometry", self.saveGeometry())	
121			event.accept()	
122			event.accept()	
124		def	get_samp_rate(self):	
125			return self.samp rate	
126				
127		def	set samp rate(self, samp rate):	
128			self.samp rate = samp rate	
			<pre>self.samp rate = samp rate self.uhd_usrp_sink_1.set_samp_rate(self.samp_rate)</pre>	
128			self.samp rate = samp rate	
128 129			<pre>self.samp rate = samp rate self.uhd_usrp_sink_1.set_samp_rate(self.samp_rate)</pre>	
128 129 130 131 132			<pre>self.samp rate = samp rate self.uhd_usrp_sink_1.set_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate)</pre>	
128 129 130 131 132 133	- , cla		<pre>self.samp rate = samp rate self.uhd_usrp_sink_1.set_samp_rate(self.samp_rate)</pre>	
128 129 130 131 132 133 134	- Cla	ass te	<pre>self.samp rate = samp rate self.uhd_usrp_sink_1.set_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) ext_rx(gr.top_block, Qt.QWidget):</pre>	
128 129 130 131 132 133 134 135	⊖ ⊽ cla	ass te	<pre>self.samp rate = samp rate self.uhd_usrp_sink 1.set_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) ext_tx(gr.top_block, Qt.OWidget): </pre>	
128 129 130 131 132 133 134 135 136	- ⊽ cla	ass te def	<pre>self.samp rate = samp rate self.uhd_usrp_sink_1.set_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) ext_rx(gr.top_block, Qt.QWidget): </pre>	
128 129 130 131 132 133 134 135 136 137	- cla	ass te def	<pre>self.samp rate = samp rate self.uhd_usrp_sink_1.set_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) ext_rx(gr.top_block, Qt.OWidget): </pre>	
128 129 130 131 132 133 134 135 136 137 138	⊂ ⊂ cla	ass te def	<pre>self.samp rate = samp rate self.uhd_usrp_sink_1.set_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) ext_rx(gr.top_block, Qt.QWidget): init(self): gr.top_blockinit(self, "Text Rx") Qt.QWidgetinit(self) self.setWindowTitle("Text Rx")</pre>	
128 129 130 131 132 133 134 135 136 137 138 139	cla	ass te def	<pre>self.samp rate = samp rate self.uhd_usrp_sink 1.set_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) ext_rx(gr.top_block, Qt.OWidget): </pre>	
128 129 130 131 132 133 134 135 136 137 138 139 140	- cla	ass te def	<pre>self.samp rate = samp rate self.uhd_usrp_sink_1.set_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) ext_rx(gr.top_block, Qt.QWidget): init(self): gr.top_blockinit(self, "Text Rx") Qt.QWidgetinit(self) self.setWindowTitle("Text Rx") qtgui.util.check_set_qss() try:</pre>	
128 129 130 131 132 133 134 135 136 137 138 139 140 141	cla v	ass te def	<pre>self.samp rate = samp rate self.uhd_usrp_sink_1.set_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) ext_rx(gr.top_block, Qt.QWidget): </pre>	
128 129 130 131 132 133 134 135 136 137 138 139 140 141 142	- cla	ass te def	<pre>self.samp rate = samp rate self.uhd_usrp_sink 1.set_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) ext_rx(gr.top_block, Qt.QWidget): </pre>	
128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143	⊂ cla	def	<pre>self.samp rate = samp rate self.uhd_usrp_sink_1.set_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) ext_rx(gr.top_block, Qt.OWidget): init(self): gr.top_blockinit(self, "Text Rx") Qt.QWidgetinit(self) self.setWindowTite("Text Rx") qtgui.util.check_set_qss() try: self.setWindowIcon(Qt.QIcon.fromTheme('gnuradio-gro')) except: pass</pre>	
128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144	⊂ cla	def	<pre>self.samp rate = samp rate self.uhd_usrp_sink_1.set_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) ext_rx(gr.top_block, Qt.QWidget): init(self): gr.top_blockinit(self, "Text Rx") Qt.QWidgetinit(self) self.setWindowTitle("Text Rx") qtgui.util.check_set_qss() try: self.setWindowIcon(Qt.QICon.fromTheme('gnuradio_grc')) except: pass self.setJayout = Qt.QVBoxLayout()</pre>	
128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145	⊂ cla	def	<pre>self.samp rate = samp rate self.uhd_usrp_sink_1.set_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) ext_rx(gr.top_block, Qt.QWidget): </pre>	
128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146	⊂ cla	def	<pre>self.samp rate = samp rate self.uhd_usrp_sink_1.set_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) ext_rx(gr.top_block, Qt.QWidget): init(self): gr.top_blockinit(self, "Text Rx") Qt.QWidgetinit(self) self.setWindowTitle("Text Rx") qtgui.util.check_set_qss() try: self.setWindowIcon(Qt.QICon.fromTheme('gnuradio_grc')) except: pass self.setJayout = Qt.QVBoxLayout()</pre>	
128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145		def	<pre>self.samp rate = samp rate self.uhd_usrp_sink_1.set_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) ext_rx(gr.top_block, Qt.OWidget): init(self): gr.top_blockinit(self, "Text Rx") Qt_OWidgetinit(self) self.setWindowTitle("Text Rx") qtgui.util.check_set_qss() try: self.setWindowIcon(Qt.QIcon.fromTheme('gnuradio-grc')) except: pass self.top_scroll_layout = Qt.QVBoxLayout() self.setLayout(self.top_scroll_layout) self.setLayout(self.top_scroll_layout) self.top_scroll = Qt.QVBoxLayout() self.top_scroll.setFrameStyle(Qt.QFrame.NoFrame)</pre>	
128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 144 145	⊂ ola	def	<pre>self.samp rate = samp rate self.uhd_usrp_sink 1.set_samp_rate(self.samp_rate) self.uhd_usrp_sink 1.set_sample_rate(self.samp_rate) ext_rx(gr.top_block, Qt.QWidget): init(self): gr.top_blockinit(self, "Text Rx") Qt.QWidgetinit(self) self.setWindowTitle("Text Rx") qtgui.util.check_set_gss() try: self.setWindowIcon(Qt.QIcon.fromTheme('gnuradio-grc')) except: pass self.top_scoll_layout = Qt.QVBoxLayout() self.setLayout(self.top_scroll_layout) self.top_scoll = Qt.QSrollAres()</pre>	
128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 144 145 144 145	₽ ₽ cla	def	<pre>self.samp rate = samp rate self.uhd_usrp_sink 1.set_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) ext_rx(gr.top_block, Qt.QWidget): </pre>	
128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148	9 9 cla	def	<pre>self.samp rate = samp rate self.uhd_usrp_sink_1.set_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) ext_rx(gr.top_block, Qt.QWidget): init(self): gr.top_blockinit(self, "Text Rx") Qt.QWidgetinit(self) self.setWindowTitle("Text Rx") qtgui.util.check_set_qss() try: self.setWindowIcon(Qt.QIcon.fromTheme('gnuradio-gre')) except: pass self.setWindowIcon(Qt.QVBoxLayout() self.setLayout(self.top_scroll_layout) self.setLayout(self.top_scroll_layout) self.top_scroll = Qt.QVPoxLayout() self.top_scroll.setFrameStyle(Qt.QFrame.NoFrame) self.top_scroll_layout.addWidget(self.top_scroll)</pre>	
128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150	9 9 cla 9	def	<pre>self.samp rate = samp rate self.uhd_usrp_sink_1.set_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) ext_rx(gr.top_block, Qt.OWidget): init(self): gr.top_blockinit(self, "Text Rx") Qt.OWidgetinit(self) self.setWindowTitle("Text Rx") qtgui.util.check_set_qss() try: self.setWindowIcon(Qt.OIcon.fromTheme('gnuradio-gro')) except: pass self.top_scroll_layout = Qt.OWBoxLayout() self.setLayout(self.top_scroll_layout) self.setFrameStyle(Qt.OFrame.NoFrame) self.top_scroll_setFrameStyle(Qt.OFrame.NoFrame) self.top_scroll_setFrameStyle(Qt.OFrame.NoFrame) self.top_scroll_setFrameStyle(Qt.OFrame.NoFrame) self.top_scroll_setFrameStyle(Qt.OFrame.NoFrame) self.top_scroll_setFrameStyle(Qt.OFrame.NoFrame) self.top_scroll_setFrameStyle(Qt.OFrame.NoFrame) self.top_scroll_setFrameStyle(Qt.OFrame.NoFrame) self.top_scroll_setFrameStyle(Qt.OFrame.NoFrame) self.top_scroll_setFrameStyle(Qt.OFrame.NoFrame) self.top_scroll_setFrameStyle(Qt.OFrame.NoFrame) self.top_scroll_setFrameStyle(Qt.OFrame.NoFrame) self.top_scroll_setFrameStyle(Qt.OFrame.NoFrame) self.top_widget = Qt.OWidget() self.top_widget = Qt.OWidget() self.top_widget = Qt.OWidget() self.top_scroll_setFrameStyle(Qt.OFrame.NoFrame) self.top_scroll_setFrameStyle(Qt.OFrame.NoFrame) self.top_widget = Qt.OWidget() self.top_widget = Qt.OWidget() self.top_scroll_setFrameStyle(Qt.OFrame.NoFrame) self.top_widget = Qt.OWidget() self.top_scroll_setFrameStyle(Qt.OFrame.NoFrame) self.top_scroll_setFrameStyle(Qt.OFrame.NoFrame) self.top_scroll_setFrameStyle(Qt.OFrame.NoFrame) self.top_widget = Qt.OWidget() setFrameStyle(Qt.OFrame.NoFr</pre>	
128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 145 145 145 145 145 145 149 150	₽ ⊽cla ⊽	def	<pre>self.samp rate = samp rate self.uhd_usrp_sink_1.set_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) ext_rx(gr.top_block, Qt.QWidget): init(self): gr.top_blockinit(self, "Text Rx") Qt.QWidgetinit(self) self.setWindowTitle("Text Rx") qtyui.util.check_set_gss() try: self.setWindowIcon(Qt.QIcon.fromTheme('gnuradio-grc'))) except: pass self.setBindowIcon(Qt.QUcon.fromTheme('gnuradio-grc')) except: pass self.setLayout(self.top_scroll_layout) self.setLayout(self.top_scroll_layout) self.setLayout(self.top_scroll_layout) self.setLayout(self.top_scroll_layout) self.top_scroll.setTrameStyle(Qt.QTexne) self.top_scroll.setTrameStyle(Qt.QTexne) self.top_scroll.setTrameStyle(Qt.QTexne) self.top_scroll.setTrameStyle(Qt.SetI.top_scroll) self.top_scroll.setTrameStyle(Qt.SetI.top_scroll) self.top_scroll.setWidgetResizable(True) self.top_scroll.setWidget() self.top_scroll.setWidget() self.top_scroll.setWidget(self.top_widget)</pre>	
128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 144 145 146 147 148 150 151		def	<pre>self.samp rate = samp rate self.uhd_usrp_sink 1.set_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) ext_rx(gr.top_block, Qt.QWidget): </pre>	
128 129 130 131 132 133 134 135 136 137 138 139 140 141 141 145 144 145 144 145 150 151 152 153 154	₽ ₽ cla	def	<pre>self.samp rate = samp rate self.uhd_usrp_sink_1.sec_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) ext_rx(gr.top_block, Qt.QWidget): init(self): gr.top_blockinit(self, "Text Rx") Qt.QWidgetinit(self, "Text Rx") Qt.QWidgetinit(self) self.setWindowTitle("Text Rx") qtgui.util.check_set_qss() try: self.setWindowTitle("Text Rx") qtgui.util.check_set_qss() try: self.setWindowTitle(Dev.Gov.fromTheme('gnuradio-gro')) except: pass self.top_scroll_layout = Qt.QWoxLayout() self.setLayout(self.top_scroll_layout) self.setLayout(self.top_scroll_ayout) self.top_scroll_setFrameStyle(Qt.QFrame.NoFrame) self.top_scroll_setFrameStyle(Qt.QFrame.NoFrame) self.top_scroll_setFrameStyle(self.top_scroll) self.top_scroll_setWidget(self.top_scroll) self.top_scroll_setWidget(self.top_widget) self.top_arcll.setWidget(self.top_widget) self.top_grid_layout = Qt.QWoxLayout(self.top_widget) self.top_grid_layout = Qt.QWoxLayout(self.top_widget) self.top_grid_layout = Qt.QWidget() self.top_grid_layout = Q</pre>	
128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 144 145 151 152 153	9 9 9	def	<pre>self.samp rate = samp rate self.uhd_usrp_sink_1.sec_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) ext_rx(gr.top_block, Qt.QWidget): init(self): gr.top_blockinit(self, "Text Rx") Qt.QWidgetinit(self, "Text Rx") Qt.QWidgetinit(self) self.setWindowTitle("Text Rx") qtgui.util.check_set_qss() try: self.setWindowTitle("Text Rx") qtgui.util.check_set_qss() try: self.setWindowTitle(Dev.Gov.fromTheme('gnuradio-gro')) except: pass self.top_scroll_layout = Qt.QWoxLayout() self.setLayout(self.top_scroll_layout) self.setLayout(self.top_scroll_ayout) self.top_scroll_setFrameStyle(Qt.QFrame.NoFrame) self.top_scroll_setFrameStyle(Qt.QFrame.NoFrame) self.top_scroll_setFrameStyle(uself.top_scroll) self.top_scroll_setWidget(self.top_scroll) self.top_scroll_setWidget(self.top_widget) self.top_arcll.setWidget(self.top_widget) self.top_grid_layout = Qt.QWoxLayout(self.top_widget) self.top_grid_layout = Qt.QWoxLayout(self.top_widget) self.top_grid_layout = Qt.QWidget() self.top_grid_layout = Qt.QWi</pre>	
128 129 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 146 151 152 153 154 155	₽ ♥ cla	def	<pre>self.samp rate = samp rate self.uhd_usrp_sink_1.set_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) ext_rx(gr.top_block, Qt.QWidget): init(self): gr.top_blockinit(self, "Text Rx") Qt.QWidgetinit(self) self.setWindowTitle("Text Rx") qtgui.util.check_set_qss() try: self.setWindowIcon(Qt.QIcon.fromTheme('gnuradio-gre')) except: pass self.top_scroll_layout = Qt.QVBoxLayout() self.setWindowIcon(Qt.QICon.fromTheme('gnuradio-gre')) except: pass self.top_scroll_layout = Qt.QVBoxLayout() self.setLayout(self.top_scroll_layout) self.top_scroll_setTrameStyle(Qt.QFrame.NoFrame) self.top_scroll_setTrameStyle(Qt.QFrame.NoFrame) self.top_scroll.setWidgetResizable(True) self.top_scroll.setWidgetResizable(True) self.top_scroll.setWidget(self.top_widget) self.top_scroll.setWidget(self.top_widget) self.top_layout = Qt.QWBoxLayout() self.top_layout = Qt.QWBoxLayout() self.top_layout = Qt.QWBoxLayout(self.top_widget) self.top_ard_layout = Qt.QWBoxLayout() self.top_ard_layout = Qt.QWBoxLayout() self.top_grid_layout = Qt.QWBoxLayout(self.top_widget) self.top_layout.addLayout(self.top_grid_layout) self.top_layout.addLayout(self.top_grid_layout)</pre>	
1299 1299 1300 1311 1323 1334 1355 1366 1377 1380 1397 1380 1397 1410 1411 1422 1433 1444 1455 1444 1455 1551 1554 1555 1556	e ♥ola	def	<pre>self.samp rate = samp rate self.uhd_usrp_sink_1.sec_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) ext_rx(gr.top_block, Qt.QWidget): init(self): gr.top_blockinit(self, "Text Rx") Qt.QWidgetinit(self, "Text Rx") Qt.QWidgetinit(self, "Text Rx") Qt.QWidgetinit(self, "Text Rx") qtgui.util.check_set_qss() try: self.setWindowTicle("Text Rx") gtss self.setWindowTicle(Qt.QIcon.fromTheme('gnuradio-gro')) except: pass self.top_scroll_layout = Qt.QWoxLayout() self.setLayout(self.top_scroll_layout) self.setLayout(self.top_scroll_layout) self.top_scroll = Qt.QScrollArea() self.top_scroll.setFrameStyle(Qt.QFrame.NoFrame) self.top_scroll.setWidgetResizable(True) self.top_scroll.setWidgetResizable(True) self.top_scroll.setWidgetResizable(True) self.top_scroll.setWidgetResizable(True) self.top_scroll.setWidget(self.top_widget) self.top_grid_layout = Qt.QWidse() self.top_grid_layout = Qt.QGridLayout() self.top_layout = Qt.QWidse(self.top_widget) self.top_layout = Qt.QWidse(self.top_widget)</pre>	
1289 1299 1300 1311 132 1333 1344 1355 1367 1380 1400 1411 1422 1433 1444 1455 1465 1474 1489 1500 1512 1553 1554 1555 1557	₽ ₽ ₽	def	<pre>self.samp rate = samp rate self.ubd_usrp_sink_1.set_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) ext_rx(gr.top_block, Qt.QWidget): </pre>	
128 129 130 131 132 133 134 135 137 138 139 141 142 143 144 145 146 147 148 150 151 152 153 154 155 156 157 158 159 160	₽	def	<pre>self.samp rate = samp rate self.uhd_usrp_sink_1.sec_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) ext_rx(gr.top_block, Qt.QWidget): init(self): gr.top_blockinit(self, "Text Rx") Qt.QWidgetinit(self, "Text Rx") Qt.QWidgetinit(self, "Text Rx") Qt.QWidgetinit(self, "Text Rx") qtgui.util.check_set_qss() try: self.setWindowTicle("Text Rx") gtss self.setWindowTicle(Qt.QIcon.fromTheme('gnuradio-gro')) except: pass self.top_scroll_layout = Qt.QWoxLayout() self.setLayout(self.top_scroll_layout) self.setLayout(self.top_scroll_layout) self.top_scroll = Qt.QScrollArea() self.top_scroll.setFrameStyle(Qt.QFrame.NoFrame) self.top_scroll.setWidgetResizable(True) self.top_scroll.setWidgetResizable(True) self.top_scroll.setWidgetResizable(True) self.top_scroll.setWidgetResizable(True) self.top_scroll.setWidget(self.top_widget) self.top_grid_layout = Qt.QWidse() self.top_grid_layout = Qt.QGridLayout() self.top_layout = Qt.QWidse(self.top_widget) self.top_layout = Qt.QWidse(self.top_widget)</pre>	
128 129 129 130 131 132 133 134 134 135 137 136 137 139 141 142 143 144 145 146 149 151 152 155 155 156 157 156 157 156 150 160 1601 161		def	<pre>solf.samD_rate = samp_rate solf.ubd_usrp_sink_1.set_samp_rate(solf.samp_rate) solf.blocks_throttle_0.set_sample_rate(solf.samp_rate) ext_rx(gr.top_block, Ot.QWidget): </pre>	
129 129 129 130 131 132 133 134 135 136 137 138 136 141 142 143 144 145 144 145 144 145 151 152 153 154 155 156 157 159 160 159 161 161		def	<pre>self.samD rate = samm_rate self.ubd_usrp_sink_1.set_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) ext_rx(gr.top_block, Qt.QWidget): int(self): gr.top_blockint(self, "Text Rx") Qt.QWidgetint(self) self.setWindowTite("Text Rx") qtgui.util.check_set_qss() try: self.setWindowTice("text Rx") qtgui.util.check_set_qss() try: self.setWindowTice("text Rx") except: pass self.top_scroll_layout = Qt.QVBoxLayout() self.setLayout(self.top_scroll_layout) self.top_scroll_setTextResttyle(Ct.OFrame.NoFrame) self.top_scroll_setWidget(self.top_scroll) self.top_scroll_setWidget(self.top_scroll) self.top_scroll_setWidget(self.top_scroll) self.top_scroll_setWidget(self.top_widget) self.top_scroll_setWidget(self.top_widget) self.top_scroll_setWidget(self.top_widget) self.top_scroll_setWidget(self.top_widget) self.top_layout = Qt.QVBoxLayout(self.top_widget) self.top_layout = Qt.QVBoxLayout(self.top_widget) self.top_layout = Qt.QVBoxLayout(self.top_widget) self.top_layout = Qt.QVBoxLayout(self.top_widget) self.top_layout.addLayout(self.top_grid_layout) self.setTings = Qt.QSetTings("GNU Radio", "text_rx") self.setTings = Qt.QSetTings("GNU Radio", "text_rx") self.setTings("GNU Radio", "text_rx") self.setTings("GNU Radio", "text_rx") self.setTings("GNU Radio", "text_rx") self.setTings("GNU Radio", "text_rx") self.setTings("GNU Radio", "text_rx") self.setTings("GNU Radio", "text_</pre>	
128 1229 1229 1230 121 133 134 135 137 138 139 141 142 143 144 145 146 147 148 150 151 152 153 154 155 156 159 160 161 162 163	 ₽ ₽ ₽ ₽ ₽ 	def	<pre>solf.samD_rate = samp_rate solf.ubd_usrp_sink_1.set_samp_rate(solf.samp_rate) solf.blocks_throttle_0.set_sample_rate(solf.samp_rate) ext_rx(gr.top_block, Ot.QWidget): </pre>	
129 129 129 130 131 132 133 134 135 136 137 138 136 141 142 143 144 145 144 145 144 145 151 152 153 154 155 156 157 159 160 159 161 161	₽	def	<pre>self.samD rate = samm_rate self.ubd_usrp_sink_1.set_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) ext_rx(gr.top_block, Qt.QWidget): int(self): gr.top_blockint(self, "Text Rx") Qt.QWidgetint(self) self.setWindowTite("Text Rx") qtgui.util.check_set_qss() try: self.setWindowTice("text Rx") qtgui.util.check_set_qss() try: self.setWindowTice("text Rx") except: pass self.top_scroll_layout = Qt.QVBoxLayout() self.setLayout(self.top_scroll_layout) self.top_scroll_setTextResttyle(Ct.OFrame.NoFrame) self.top_scroll_setWidget(self.top_scroll) self.top_scroll_setWidget(self.top_scroll) self.top_scroll_setWidget(self.top_scroll) self.top_scroll_setWidget(self.top_widget) self.top_scroll_setWidget(self.top_widget) self.top_scroll_setWidget(self.top_widget) self.top_scroll_setWidget(self.top_widget) self.top_layout = Qt.QVBoxLayout(self.top_widget) self.top_layout = Qt.QVBoxLayout(self.top_widget) self.top_layout = Qt.QVBoxLayout(self.top_widget) self.top_layout = Qt.QVBoxLayout(self.top_widget) self.top_layout.addLayout(self.top_grid_layout) self.setTings = Qt.QSetTings("GNU Radio", "text_rx") self.setTings = Qt.QSetTings("GNU Radio", "text_rx") self.setTings("GNU Radio", "text_rx") self.setTings("GNU Radio", "text_rx") self.setTings("GNU Radio", "text_rx") self.setTings("GNU Radio", "text_rx") self.setTings("GNU Radio", "text_rx") self.setTings("GNU Radio", "text_</pre>	

166	# Blocks
167	
168	<pre>self.uhd_usrp_source_0 = uhd.usrp_source(</pre>
169	",".join(("", "")),
170	uhd.stream args (
171	dial-of-clan_relg-(cpu format="fo32",
172	<pre>channels=range(1),</pre>
173).
174)
175	self.uhd_usrp_source_0.set_samp_rate(samp_rate)
176	<pre>self.uhd_usrp_source_0.set_center_freq(1000000000, 0)</pre>
177	<pre>self.uhd_usrp_source_0.set_gain(30, 0)</pre>
178	<pre>self.uhd_usrp_source_0.set_antenna('RX2', 0)</pre>
179	self.gtgui_freq_sink_x_0 = qtgui.freq_sink_c(
180	1024, #size
181	firdes.WIN BLACKMAN hARRIS, #wintype
182	0, #fg
183	samp rate, #bw
184	"", frame
185	1 #number of inputs
186	I when be of inputs
) 15 start for sich a 0 oct webbe time (0.10)
187	self.qtgui_freq_sink_x_0.set_update_time(0.10)
188	self.qtgui_freq_sink_x_0.set_y_axis(-140, 10)
189	<pre>self.qtgui_freq_sink_x_0.set_y_label('Relative Gain', 'dB')</pre>
190	<pre>self.qtgui_freq_sink_x_0.set_trigger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "")</pre>
191	<pre>self.qtgui_freq_sink_x_0.enable_autoscale(False)</pre>
192	self.qtgui_freq_sink_x_0.enable_grid(False)
193	<pre>self.qtgui_freq_sink_x_0.set_fft_average(1.0)</pre>
194	self.qtgui_freq_sink_x_0.enable_axis_labels(True)
195	self.qtgui_freq_sink_x_0.enable_control_panel(False)
196	
197	if not True:
198	<pre>self.qtgui_freq_sink_x_0.disable_legend()</pre>
199	
200	<pre>if "complex" == "float" or "complex" == "msg_float":</pre>
201	self.qtgui_freq_sink_x_0.set_plot_pos_half(not True)
202	
203 🤤	labels = ['', '', '', '', '',
204	
205 🖯	widths = [1, 1, 1, 1, 1, 1,
206	1, 1, 1, 1, 1]
207 🖯	colors = ["blue", "red", "green", "black", "cyan",
208	"magenta", "yellow", "dark red", "dark green", "dark blue"]
209	alphas = [1.0, 1.0, 1.0, 1.0, 1.0,
210	1.0, 1.0, 1.0, 1.0, 1.0
211	for in xrange(1):
212	<pre>if landlabels(i) == 0:</pre>
213	
213	<pre>self.qtgui_freq_sink_x_0.set_line_label(i, "Data {0}".format(i)) else:</pre>
214	esse: self.qtgui freq sink x 0.set line label(i, labels[i])
216	solf.qtpui_freq_sink_O.set line_width(i, Musters(i)) self.qtpui_freq_sink_0.set line_width(i, widths[i])
217	
218	self.qtgui_freq_sink_x_0.set_line_color(i, colors[i]) self.qtgui_freq_sink_x_0.set_line_alpha(i, alphas[i])
219	Serriggar_reg_Sink_x_0.see_rine_arpha(r, arphas(r))
220	self. <u>qtgui</u> freq_sink x 0 win = sip.wrapinstance(self.qtgui freq_sink x 0.pyqwidget(), Qt.QWidget)
221	<pre>self.top_grid_layout.addWidget(selfgtgui_freq_sink_x_0_win)</pre>
222	<pre>self.low_pass_filter_0 = filter.fir_filter_ccf(1, firdes.low_pass(</pre>
223	1, samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76))
224	self.digital_gmsk_demod_0 = digital.gmsk_demod(
225	samples_per_symbol=2,
226	gain_mu=0.175,
227	mu=0.5,
228	<pre>omega_relative_limit=0.005,</pre>
229	freq_error=0.0,
230	verbose=False,
231	log=False,
232)
233	
234	<pre>clr_file = open('/home/c/test_file2', 'w')</pre>
235	clr_file.close()
236	
237	<pre>self.blocks_multiply_const_vxx_0 = blocks.multiply_const_vcc((1,))</pre>
238	<pre>self.blocks_file_sink_0 = blocks.file_sink(gr.sizeof_char*1, '/home/c/test_file2', False)</pre>
239	self.blocks file sink 0.set unbuffered(False)

<pre>interested in the second second</pre>	240			self. <u>blks2_packet_decoder_0 = grc_blks2.packet_demod_b(grc_blks2.packet_decoder(</u>	
<pre>int control is a set of a</pre>					
<pre>statuselpace a, payled: allowedpace.peergitid, payled;</pre>					
<pre>int</pre>	243				
<pre>int</pre>	244				
<pre>int of intermediate interm</pre>	245)	
<pre>intermed to the second se</pre>					
<pre>provide state of the set of</pre>					
<pre>provide set of the set of th</pre>					
<pre>11 / Concections 12 / Concections 13 / Concections 14 / Concections 15 / Concections 15 / Content((iii:Listaguart_Genet(, 0, 0), (calf.biots_file_pist(, 0)) 15 / Content((ii:Listaguart_Genet(, 0))) 15 / Cont</pre>					
<pre>provide set in the set is th</pre>					
<pre>std stl.compet(usl.klash_packet_decoder_0, 0), (ssl.klash_file_isid_0, 0) stl.compet(usl.klash_packet_decoder_0, 0), (ssl.klash_file_isid_0, 0) stl.compet(usl.klash_packet_decoder_0, 0), (ssl.klash_packet_decoder_0, 0) stl.compet(usl.compact_filer_0, 0), (ssl.klash_packet_decoder_0, 0) stl.compet(usl.compact_filer_0, 0), (ssl.thash_packet_decoder_0, 0) stl.compet(usl.compact_filer_0, 0), (ssl.thash_packet_decoder_0, 0) stl.compet(usl.compact_filer_0, 0), (ssl.thash_packet_decoder_0, 0) stl.compet(usl.compact_filer_0, 0), (ssl.thash_packet_decoder_0, 0) stl.compet(usl.compact_filer_0, 0), (ssl.thash_packet_0, 0) stl.compet(usl.compact_filer_0, 0), (ssl.thash_packet_0, 0) stl.compet(usl.compact_filer_0, 0) stl.compet(usl.compet(usl.compet(usl.compet(usl.comp_tasket_0, 0)) stl.compet(usl.compet(usl.compet(usl.compet(usl.comp_tasket_0, 0)) stl.compet(usl.compet(usl.compet(usl.comp_tasket_0, 0)) stl.compet(usl.compet(usl.compet(usl.comp_tasket_0, 0)) stl.compet(usl.compet(usl.comp_tasket_0, 0)) stl.compet(usl.comp_tasket_0, 0) stl.compet(usl.comp_tasket_0, 0) stl.compet(usl.compet(usl.compet(usl.compet(usl.comp_tasket_0, 0)) stl.compet(usl.compet(usl.compet(usl.compet(usl.comp_tasket_0, 0)) stl.compet(usl.compet(</pre>					
<pre>self consect(esif_block_multiply_cons_vest(esif_block_school, e)) self.consect(esif_block_max_filer_0, e)(self_block_school, e)) self.consect(esif_block_max_filer_0, e), (self_schi_typu_fest_school, e)) self.consect(esif_block_max_filer_0, e), (self_schi_typu_fest_school, e)) self.consect(esif_block_max_filer_0, e), (self_schi_typu_fest_school, e)) self.consect(esif_block_max_filer_e), e), (self_schi_typu_fest_schi_tx_0, e)) self.consect(esif_block_max_filer_e), e), (self_schi_typu_fest_schi_tx_0, e)) self.consect(esif_block_max_filer_e), e), (self_schi_typu_fest_schi_tx_0, e)) self.consect(esif_block_max_filer_e), e); (self_schi_typu_fest_schi_tx_0, e)) self.consect(esif_block_max_filer_e), esif_schi_typu_fest_schi_tx_0, e)) self.consect(esif_block_max_filer_e), esif_schi_typu_fest_schi_tx_0, e)) self_schi_typu_fest_schi_typu_fe</pre>	253				
<pre>shi connect(shi:logsas_filer_0, 0), (shi:digita_mak_demod_0, 0)) shi connect((shi:logsas_filer_0, 0), (shi:diguta_mak_demod_0, 0)) shi connect((shi:logsas_filer_0, 0), (shi:diguta_mak_demod_0, 0)) shi connect((shi:logsas_filer_0, 0), (shi:loka_multiply_const_ws_0, 0)) dif disemption (shi:logsas_filer_0, 0), (shi:loka_multiply_const_ws_0, 0), (shi:loka</pre>	254				
<pre>shi commet((=shi(:=grave_set_site_0, 0), (=shi(spit_(=grave_site_g, 0))) shi shi commet((=shi(:=grave_0, 0), (=shi(sbi(=grave_site_g, 0))) def sitestigs = 0:(0etting('=grave_o, 0), (=shi(sbi(=grave_o, 0)))) def sitestigs = 0:(0etting('=grave_o, 0))) def sitestigs = 0:(0etting('=grave_o, 0))) def sitestigs = 0:(0etting('=grave_o, 0))) def sitestigs = 0:(0etting('=grave_o</pre>					
<pre>self.commect((welf.ubd_usrp_source_0, 0), (welf.blocks_multiply_const_usr_0, 0)) def closeftrent(welf, event): eelf.commect((welf.ubd_usrp_source_0, 0), (welf.blocks_multiply_const_usr_0, 0)) eelf.setting= C0.Getting(('OUW Redio', 'text_pr') eelf.setting= C0.Getting(('OUW Redio', 'text_pr') eelf.setting= C0.Getting('OUW Redio', 'text_pr') for.Getting= C0.Getting('OUW Redio', 'text_pr') eelf.setting= C0.Getting('OUW Redio', 'text_pr') eelf.setting= C0.Getting('OUW COUW Redio', 'text_pr') eelf.setting= C0.Getting('OUW COUW Redio', 'text_pr') eelf.setting= C0.Getting('OUW COUW COUW) eelf.setting= C0.Getting('OUW COUW) eelf.setting= C0.Getting('OUW COUW) eelf.setting= C0.Getting('OUW Redio', 'text') weipt_file = Goe('OUW Addov', 'text') weipt_file.setting= C</pre>					
<pre>def closeTrent(sif, even):</pre>					
<pre>def classYvant(wil, event):</pre>				<pre>seif.connect((seif.uhd_usrp_source_0, 0), (seif.blocks_multiply_const_vxx_0, 0))</pre>	
<pre>pid of clearcont(sil, vers):</pre>					
<pre>set: set: set: set: set: set: set: set:</pre>			def	<pre>closeEvent(self, event):</pre>	
<pre>sel:.setLings.setVile('geometry', self.areGemetry()) event.scopt() def get_manp_rate(self): return self.samp_rate(self.samp_rate) self.samp_rate(self.samp_rate): self.samp_rate = asspr_rate(self.samp_rate): self.samp_rate = samp_rate(self.samp_rate): self.samp_rate = samp_rate = doood self.samp_rate = samp_rate = doood self.samp_rate = samp_rate(self.samp_rate): self.samp_rate = samp_rate = samp_rate; "self.samp_rate): self.samp_rate = samp_rate = doood sel</pre>					
<pre>idf get_manp_rate(sif): return solf.samp_rate idf get_manp_rate(sif): return solf.samp_rate idf get_manp_rate(sif): return solf.samp_rate idf get_manp_rate(sif): return solf.samp_rate(sif): return solf.samp_rate(sif): solf.samp_rate(sif): return solf.samp_rate(sif): solf.samp_rate(sif): return solf.samp_rate(sif): solf.samp_rate(sif): return solf.samp_rate(sif):</pre>	263				
<pre>def get_samg_stat(s1); rstum molf.samp_state def def_samg_rate(s1, samp_rate); rsti.samp_rate = samp_rate esif.udu_ursout0.cost_samp_rate(s1, samp_rate) esif.udu_ursout0.cost_samp_rate(s1, samp_rate) esif.udu_ursout0.cost_samp_rate(s1, samp_rate) esif.udu_ursout0.cost_samp_rate(s1, samp_rate, 20000), 50000, firdes.WIN_WANDENE, 6.76)) defsint(salf); gr:top_block, 0c.Offidget):</pre>	264				
<pre>d</pre>					
<pre>def est_samp_rate(=:</pre>					
<pre>def set_amp_rate(self, smp_rate):</pre>				return self.samp_rate	
<pre>101 self.asm_rate = samp_rate 102 self.asm_rate = samp_rate 102 self.asm_tarp_ource_orset_amp_rate(self.asmp_rate) 102 self.low_pass_filter_0.set_amp(fides.low_pass(i, self.asmp_rate, 200000, 50000, firdes.WIN_HANDING, 6.76)) 103 definit(self): 104 definit(self): 105 definit(self): 105 definit(self): 105 definit(self): 106 definit(self): 107 definit(self): 108 definit(self): 109 definit(self): 109 definit(self): 100 self.setWindwrion(Cdt.OIcon.fromTheme('gnuradle-gro')) 100 self.setWindwrion(Cdt.OIcon.fromTheme('gnuradle-gro')) 101 self.setWindwrion(Cdt.OIcon.fromTheme('gnuradle-gro')) 102 self.setWindwrion(Cdt.OIcon.fromTheme('gnuradle-gro')) 103 self.top_serColl_astwride(self.top_serColl) 104 self.top_serColl_astwride(self.top_gro')) 105 self.setWindwride(self.top_gro') 105 self.setWindwride(self.top_gro')) 106 self.top_coroll.setWindwride(self.top_gro')) 107 self.setWindwride(self.setWindwr) 108 self.top_layout.addWindwr)."obyteArray()) 108 self.top_layout.addWindwr)."self.setWindwride(self.setWindwr) 109 self.setWindwride(self.setWindwr)."self.setWindwride(self.setWindwr) 100 self.setWindwride(self.setWindwr)."self.setWindwr) 100 self.setWindwride(self.setWindwr)."self.setWindwr) 100 self.setWindwride(self.setWindwr)."self.setWindwr) 100 self.setWindwride(self.setWindwr)."self.setWindwr) 100 self.setWindwride(self.setWindwr) 100 self.setWindwride(self.setWindwr) 100 self.setWindwr) 100 self.setWindwr) 100 self.setWindwr) 100 self.se</pre>			def	est samm vate/salf_ samm vate).	
<pre>prove the set of the set of</pre>					
<pre>self.qtpii_freq_sinf_c_0.set_frequency_ange(0, self.samp_rate, 20000, 50000, firdes.WIN_HAMMING, 6.76)) self.low_pass_filter_0.set_taps(firdes.low_pass(1, self.samp_rate, 20000, 50000, firdes.WIN_HAMMING, 6.76)) self.low_pass_filter_0.set_taps(firdes.low_pass(1, self.samp_rate, 20000, 50000, firdes.WIN_HAMMING, 6.76)) self.setWindowTitle("chat") self.setWindo</pre>					
<pre>self.low_pass_filter_0.set_taps(firdes_low_pass(1, solf.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76)) self.low_pass_filter_0.set_taps(firdes_low_pass(1, solf.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76)) definit(self): gr.top_blockinit(self, "Chat") ch.Omidgetinit(self) gr.top_blockinit(self) gr.top_block_init(self) gr.top_block_set_ges() grupy solf.setWindowTiel (Chat") group_scoll_layout = Qt.OTDoxLayout() self.setWindowTiel Qt.OTDox.fromTheme('gnumedia-grc')) self.top_scoll = Qt.OGTDOXLayout() self.top_scoll_layout = Qt.OTDoxLayout() self.top_scoll_layout.addWidget(self.top_scoll) self.top_scoll_layout.addWidget(self.top_scoll) self.top_scoll_layout.addWidget(self.top_scoll) self.top_scoll_layout.addWidget(self.top_scoll) self.top_scoll_layout.addWidget(self.top_scoll) self.top_scoll_layout.addWidget(self.top_scoll) self.top_scoll_layout.addWidget(self.top_scoll) self.top_scoll_layout.addWidget(self.top_scoll) self.top_uridget = Qt.OTDOxLayout() self.top_scoll_layout.addWidget(self.top_scidget) self.top_urid_layout = Qt.OTDOxLayout() self.top_layout = Qt.OTDOxLayout() self.top_grid_layout = Qt.OTDOxLayout() self.top_grid_layout(self.top_grid_layout) self.top_grid_layout(self.top_grid_layout) self.top_layout = Qt.OTDOxLayout() self.top_layout = Qt.OTDOxLayout() self.top_grid_layout = Qt.QTDOxLayout() self.top_layout = Qt.QTDOxLayout() self.top_layout</pre>					
<pre>232 class ack(gr.top_block, Qt.QWidget): 243</pre>	273				
<pre>plase ack(gr.top_block, Qt.QHidget): definit(self): gr.top_blockinit(self, "Chat") Qt.QHidgetinit(self, "Chat") Qt.QHidgetinit(self) self.setWindowTiel("Chat") qtqui.util.oheck_set_ges() tyy: self.setWindowTon(Qt.QIcon.fromTheme('gnuradio-grc')) except: self.setWindowTon(Qt.QIcon.fromTheme('gnuradio-grc')) except: self.setWindowTon(Qt.QIcon.fromTheme('gnuradio-grc')) except: self.setWindowTon(Qt.QIcon.fromTheme('gnuradio-grc')) except: self.setWindowTon(Qt.QICon.fromTheme('gnuradio-grc')) except: self.setWindowTon(Qt.QICon.fromTheme('gnuradio-grc')) except: self.setLayout(self.top_scroll) self.setLayout(self.top_scroll) self.setLayout(self.top_scroll) self.setLayout(self.top_scroll) self.setLayout(self.top_scroll) self.top_scroll.setWindgetBeizhable(True) self.top_scroll.setWindget(self.top_widget) self.top_argrid_layout = Qt.QWindgeut(self.top_widget) self.top_layout = Qt.QWindgeut(self.top_widget) self.setLagout(self.setLingowt) self.setLings = Qt.QUetLing('GNU Radio", "chat") self.setLings = Qt.QUetLing('GNU Radio", "chat") self.setLings = Qt.QUetLing('GNU Radio", "chat") self.setLings = Qt.QUetLing('GNU Radio", "chat") self.setLings = Qt.QuetLing('Chat4', 'w') weight_file = logen('/home/o/chat4', 'w') weight_file.close() weight_file.close() self.setLine('AQNU') weight_file.close() self.setLine('AQNU') weight_file.close() setLine('AQNU') weight_file.close() setLine('AQNU') weight_file.close() setLine('AQNU') weight_file.close() setLine('AQNU') weight_file.close() setLine('AQNU') setLine('AQNU') weight_file.close() setLine('AQNU') weight_file.close()</pre>					
<pre>definit(self): gr.top_blockinit(self, "Chat") Qt.(Widgetinit(self) gelf.setKindowTite("Chat") qtpui.util.check_set_gss() try: self.setKindowTite("Chat") qtpui.util.check_set_gss() try: self.setKindowTite("Chat") qtpui.util.check_set_gss() try: self.setKindowTite("Chat") qtpui.util.check_set_gss() try: self.setKindowTite("Chat") qtpui.util.check_set_gss() try: self.setKindowTite("Chat") qtpui.util.check_set_gss() try: self.setKindowTite("Chat") self.top_scoll_layout = Ot.OVBoxLayout() self.top_scoll_layout=Ot.OVBoxLayout() self.top_scoll_setTidgetKesiable(True) self.top_scoll_setTidgetKesiable(True) self.top_scoll.setTidgetKesiable(True) self.top_grid_layout = Ot.OVBoxLayout() self.top_grid_layout = Ot.OVBoxLayout() self.top_grid_layout = Ot.OVBoxLayout() self.top_grid_layout = Ot.OVBoxLayout() self.top_layout = Ot.OVBoxLayout() self.top_grid_layout = Ot.OVBoxLayout() self.top_layout = Ot.OVBoxLayout() self.trestorGenetry(self.settings.value("geometry").toByteArray()) self.restorGenetry(self.settings.value("geometry").toByteArray()) weight_file= open('/home/Achat', 'w') weight_file= open('/home/Achat', 'w') weight_file=.clos() weight_file.eclos() weight_file.eclos() ################################</pre>					
<pre>defint(self; "Chat") defint(self, "Chat") det ("idgetint(self, "Chat") self.top_scroll_layout = (t.OVBoxLayout() self.top_scroll_layout.addWidget(self.top_scroll) self.top_scroll.setWidget(self.top_widget) self.top_actoll = (t.OVBoxLayout(self.top_widget) self.top_idget = (t.OvGetingvut() self.top_idget = (t.OvGetingvut(</pre>		cla	ss ac	ck(gr.top_block, Qt.QWidget):	
279 gr.top_blockinit(self, "Chat") 280 Gt.OWidgetinit(self) 281 self.setWindowTich("Chat") 282 dtyui.util.oheck_set_gse() 283 try: 284 self.setWindowTich(Chat") 285 self.setWindowTich(Chat") 286 pss 287 self.setWindowTich(ChatChat") 288 self.setWindowTich(ChatChat") 289 self.setWindowTich(ChatChat") 289 self.top_scroll_layout = Qt.QVDoxLayout() 289 self.top_scroll_layout.addWidget(self.top_scroll) 280 self.top_scroll.setWidget(self.top_widget) 281 self.top_scroll.setWidget(self.top_widget) 282 self.top_layout = Qt.OWIndwide(self.top_widget) 283 self.top_layout = Qt.OWIndwide(self.top_widget) 284 self.top_layout = Qt.OWIndwide(self.top_widget) 285 self.top_layout = Qt.OWIndwide(self.top_widget) 286 self.top_layout = Qt.OWIndwide(self.top_widget) 287 self.top_layout = Qt.OWIndwide(self.top_widget) 288 self.top_layout = Qt.OWIndwide(self.top_widget) 289 self.top_layout = Qt.OWIndwide(self.top_widget) </td <td></td> <td></td> <td>def</td> <td>init (self).</td> <td></td>			def	init (self).	
<pre>200 0t.0%idgetinit(self) 201 self.setWindowTitle("Chat") 202 4tyii.util.check_set_ggs() 203 2ty: 204 self.setWindowTon(Qt.QTcon.fromTheme('gnuradio-gro')) 205 205 205 205 205 205 205 205 205 205</pre>					
<pre>self.setWindowTitle("Chat") gtgui.util.check_set_ges() gtgui.util.check_set_ges() try: self.setWindowIccon(Qt.QICon.fromTheme('gnuradio_gro')) self.setWindowIccon(Qt.QICon.fromTheme('gnuradio_gro')) self.setUayout(self.top_soroll_ayout) self.setUayout(self.top_soroll_ayout) self.setUayout(self.top_soroll_ayout) self.setUayout(self.top_soroll) self.setUayout(self.top_soroll) self.top_soroll_setWindowIccon(Ct.QICon.fromTheme) self.top_soroll_setWindowIccon(Ct.QICon.fromTheme('gnuradio_gro')) self.top_layout = 0t.QIContAgout(self.top_widget) self.top_layout = 0t.QIContAgout(self.top_widget) self.top_layout = 0t.QIContAgout(self.top_widget) self.top_layout.addIayout(self.top_widget)) self.top_layout.addIayout(self.top_widget) self.top_layout.addIayout(self.top_widget)) self.top_layout.addIayout(self.top_widget) self.top_layout.addIayout(self.top_widget)) self.top_layout.addIayout(self.top_widget)) self.top_layout.addIayout(self.top_widget) self.top_layout.addIayout(self</pre>					
233 try: 234 self.setWindowIcon(Qt.OIcon.fromTheme('gnuradio-gro')) 235 exept: 246 pase 257 self.top_soroll_layout = Qt.QVBoxLayout() 258 self.setLayout(self.top_soroll_layout) 259 self.top_soroll_layout = Qt.QVBoxLayout() 250 self.top_soroll_layout(self.top_soroll) 251 self.top_soroll_layout.addWidget(self.top_soroll) 252 self.top_soroll.setWidget(self.top_widget) 253 self.top_didget = Qt.QVBoxLayout(self.top_widget) 254 self.top_grid_layout = Qt.QVBoxLayout(self.top_widget) 255 self.top_grid_layout = Qt.QVBoxLayout(self.top_widget) 256 self.top_layout.addLayout(self.top_grid_layout) 257 self.settings = Qt.QGBoxLayout(self.top_grid_layout) 258 self.top_layout.addLayout(self.top_grid_layout) 259 self.settings = Qt.QGBoxLayout(self.top_grid_layout) 250 self.settings = Qt.QGBoxLayout(self.top_schurt) 259 self.settings = Qt.QGBoxLayout(self.top_schurt) 260 self.settings.value("geometry").toByteArray(!) 271 self.settings.value("geometry").toByteArray(!) 272 sel					
<pre>self.setWindowIcon(Qt.QIcon.fromTheme('gnuradig='grc')) except: pas self.top_scroll_layout = (t.QVBoxLayout() self.top_scroll_layout = (t.QVBoxLayout() self.top_scroll=(t.QCBorcollayout) self.top_scroll=(t.QCBorcollayout) self.top_scroll_setWindpetResizable(True) self.top_scroll_layout = (t.QVBoxLayout() self.top_scroll.setWindpet(self.top_windpet) self.top_scroll.setWindpet(self.top_windpet) self.top_groll_setWindpet(self.top_windpet) self.top_groll_setWindpet(self.top_windpet) self.top_groll_setWindpet(self.top_grid_layout) self.top_grid_sout = (t.QVBoxLayout()) self.top_grid_setWindpet(self.top_windpet) self.top_scroll.setWindpet(self.top_windpet) self.top_grid_sout = (t.QVBoxLayout()) self.top_scroll.setWindpet(self.top_windpet) self.top_scroll.setWindpet(self.top_windpet) self.top_scroll.setWindpet(self.top_windpet) self.top_scroll.setWindpet(self.top_windpet) self.top_scroll.setWindpet(self.top_windpet) self.top_scroll.setWindpet(self.top_scroll) self.top_scroll_screwtny(self.top_scroll) self.top_scroll_screwtny(self.top_screwtny) self.top_screwtny(self.settings.value("geometry").toByteArray()) self.gamp_rate = gamp_rate = 400000 weight_file = open('home/o/chat4', 'w') weight_file.close() weight_file.close() } </pre>	282			<pre>qtgui.util.check_set_qss()</pre>	
285 except: 286 pass 287 self.top_scroll_layout = Ot.QVBoxLayout() 288 self.top_scroll_layout) 289 self.top_scroll_layout(e(t.Orame.NoFrame)) 281 self.top_scroll_setTramStyle(ct.Orame.NoFrame)) 282 self.top_scroll_setTramStyle(ct.Orame.NoFrame)) 283 self.top_scroll.setTramStyle(ct.Orame.NoFrame)) 284 self.top_scroll.setTramStyle(ct.Orame.NoFrame)) 285 self.top_scroll.setTramStyle(ct.Orame.NoFrame)) 286 self.top_scroll.setTramStyle(ct.Orame.NoFrame)) 287 self.top_scroll.setTramStyle(ct.Orams.NoFrame)) 288 self.top_scroll.setTramStyle(ct.Orams.NoFrame)) 289 self.top_scroll.setTramStyle(ct.Orams.NoFrame)) 286 self.top_scroll.setTramStyle(ct.Orams.NoFrame) 287 self.top_layout.addEquat(Top_widget) 288 self.top_layout = Qt.QGridLayout() 289 self.settings = Qt.OSettings("GNU Radio", "chat") 280 self.settings = Qt.QSettings("GNU Radio", "chat") 281 self.settings.value("geometry").toByteArray()) 282 self.settings.value("geometry").toByteArray()) 283 self.settings =					
286 pas 287 self.top_scroll_layout = 0t.0W5xLayout() 288 self.top_scroll_ayout.0 289 self.top_scroll_ayout.0 289 self.top_scroll_astrameStyle(0t.OFrame.NoFrame) 281 self.top_scroll_ayout.addWidgt(self.top_scroll) 282 self.top_scroll.setWidgtExesizable(True) 283 self.top_scroll.setWidgtExesizable(True) 284 self.top_ardiget = 0t.0W1dget() 285 self.top_layout.addLayout(self.top_widget) 286 self.top_ardid_ayout = 0t.0W1dgayout(self.top_widget) 287 self.top_layout.addLayout(self.top_grid_layout) 288 self.top_layout.addLayout(self.top_grid_layout) 289 self.settings = 0t.0Settings("GNU Radio", "chat") 280 self.restoreGeometry(self.settings.value("geometry").toByteArray()) 301 ####################################					
<pre>287 self.top_scroll_layout = Qt.QVBoxLayout() 288 self.top_scroll_layout = Qt.QVBoxLayout() 289 self.top_scroll=Qt.QGcrollArea() 290 self.top_scroll=Qt.QGcrollArea() 291 self.top_scroll_setWidget() 292 self.top_scroll_setWidget(self.top_scroll) 293 self.top_scroll.setWidget(self.top_widget) 294 self.top_scroll.setWidget(self.top_widget) 295 self.top_layout = Qt.QVBoxLayout(self.top_widget) 296 self.top_layout.addLayout(self.top_grid_layout) 299 self.settings = Qt.QSettings("GNU Radio", "chat") 300 self.restoreGeometry(self.settings.value("geometry").toByteArray()) 301 303</pre>					
<pre>289 self.setLayout(self.top_scroll_layout) 289 self.setLayout(self.top_scroll_ayout) 289 self.top_scroll_setVaconsetVale(C.OFrame.NoFrame) 291 self.top_scroll_layout.addWidget(self.top_scroll) 292 self.top_scroll.setWidgetResizable(True) 293 self.top_scroll.setWidget(self.top_widget) 294 self.top_acroll.setWidget(self.top_widget) 295 self.top_layout = 0t.OWBoxLayout(self.top_widget) 296 self.top_layout = 0t.OWBoxLayout(self.top_widget) 297 self.top_layout.addLayout(self.top_grid_layout) 298 self.settings = Qt.QSettings("GNV Radio", "chat") 299 self.settings = Qt.QSettings.value("geometry").toByteArray()) 301 302 303 ###################################</pre>				-	
<pre>self.top_scroll = Qt.QScrollArea() self.top_scroll.setFrameStyle(Qt.QFrame.NoFrame) self.top_scroll.setFrameStyle(Qt.QFrame.NoFrame) self.top_scroll.setWidgetResizable(True) self.top_scroll.setWidgetResizable(True) self.top_scroll.setWidget(self.top_scroll) self.top_scroll.setWidget(self.top_widget) self.top_scroll.setWidget(self.top_widget) self.top_grid_layout = Qt.QGridLayout() self.top_grid_layout.addLayout(self.top_grid_layout) self.restoreGeometry(self.settings.value("geometry").toByteArray()) self.setTings = Qt.QSetTings("GNU Radio", "chat") self.setTings = Qt.QSetTings.value("geometry").toByteArray()) self.setTings = Qt.QSetTings.value("geometry").toByteArray()) self.setTings = Qt.QSetTings.value("geometry").toByteArray()) self.setTings = gamp_rate = 400000 self.setTings.value("geometry") self.setTing.value("geometry") self.setTings.value("geometry") self.setTings.value("geometry").toByteArray()) self.setTings.value("geometry") self.setTings.value("geometry").toByteArray()) self.setTings.value("geometry").toByteArray()] self.setTings.value("geometry").toByteArray()] self.setTings.value("geometry").toByteArray()] self.setTings.value("geometry").toByteArray("geometry").toByteArray("geometry").toByteArray("geometry").toByteArray("geometry").toByteArray("geometry")</pre>					
<pre>290 self.top_scroll.setFrameStyle(Qt.QFrame.NoFrame) 291 self.top_scroll_adwidget(self.top_scroll) 292 self.top_scroll.setWidget(self.top_scroll) 293 self.top_widget = Qt.QWidget() 294 self.top_layout = Qt.QVBXLayout(self.top_widget) 295 self.top_layout = Qt.QVBXLayout(self.top_grid_layout) 297 self.top_layout.addEayout(self.top_grid_layout) 298 self.settings = Qt.QSettings("GNU Radio", "chat") 300 self.restoreGeometry(self.settings.value("geometry").toByteArray()) 301 self.restoreGeometry(self.settings.value("geometry").toByteArray()) 302 self.settings = gamp_rate = 400000 303 weight_file = open('/homma/c/chat4', 'w') 309 weight_file.close() 311 ***********************************</pre>					
<pre>291 self.top_scroll_layout.addWidget(self.top_scroll) 292 self.top_scroll.setWidgetResizable(True) 293 self.top_widget = Qt.QWidget() 294 self.top_widget = Qt.QWidget) 295 self.top_layout = Qt.QVBoxLayout(self.top_widget) 296 self.top_layout.addLayout(self.top_grid_layout) 299 self.settings = Qt.QSettings("GNV Radio", "chat") 300 self.restoreGeometry(self.settings.value("geometry").toByteArray()) 301 302 303 ###################################</pre>					
<pre>293 self.top_widget = Qt.QWidget() 294 self.top_soroll.setMidget(self.top_widget) 295 self.top_layout = Qt.QWoxLayout(self.top_widget) 296 self.top_layout = Qt.QWoxLayout() 297 self.top_layout.addLayout(self.top_grid_layout) 298 self.setTings = Qt.QSetTings("GNU Radio", "chat") 300 self.restoreGeometry(self.setTings.value("geometry").toByteArray()) 301 302 303 ###################################</pre>	291				
<pre>self.top_srcll.setWidget(self.top_widget) self.top_grid_layout = Qt.QGrdLayout(self.top_widget) self.top_layout.addLayout(self.top_grid_layout) self.setWings = Qt.QSetWings("GNV Radio", "chat") self.restoreGeometry(self.setWings.value("geometry").toByteArray()) self.restoreGeometry(self.setWings.value("geometry").toByteArray()) self.restoreGeometry(self.setWings.value("geometry").toByteArray()) self.setWings.value("geometry").toByteArray()) self.setWings.value("geometry").toByteArray("geometry").toByteArray("geometry").toByteArray("geometry").toByteArray("geometry").toByteArray("geometry").toByteArray("geometry").toByteArra</pre>	292		-	self.top_scroll.setWidgetResizable(True)	
<pre>295 self.top_layout = Qt.QVBoxLayout(self.top_widget) 296 self.top_grid_layout = Qt.QCVBoxLayout() 297 self.top_layout.addLayout(self.top_grid_layout) 298 self.settings = Qt.QSettings("GNV Radio", "chat") 300 self.restoreGeometry(self.settings.value("geometry").toByteArray()) 301 302 303 ***********************************</pre>					
<pre>296 self.top_grid_layout = Qt.QGridLayout() 297 self.top_layout.addLayout(self.top_grid_layout) 298 299 self.settings = Qt.QSettings("GNV Radio", "chat") 300 self.restoreGeometry(self.settings.value("geometry").toByteArray()) 301 302 303 ###################################</pre>					
<pre>297 self.top_layout.addLayout(self.top_grid_layout) 298 299 self.settings = Qt.QSettings("GNV Radio", "chat") 300 self.restoreGeometry(self.settings.value("geometry").toByteArray()) 301 302 303 304 4 f Variables 305 5 ###################################</pre>					
<pre>299 self.settings = Qt.QSettings("GNV Radio", "chat") 300 self.restoreGeometry(self.settings.value("geometry").toByteArray()) 302 303</pre>					
<pre>299 self.settings = Qt.QSettings("GNV Radio", "chat") 300 self.restoreGeometry(self.settings.value("geometry").toByteArray()) 301 302 303 ###################################</pre>			1	serr.cop_rayout.auumayout(serr.cop_grid_rayout)	
<pre>300 self.restoreGeometry(self.settings.value("geometry").toByteArray()) 301 302 303 ###################################</pre>				self.settings = Ot.OSettings("GNU Radio", "chat")	
<pre>301 302 303 304 4 * Variables 305 305 306 self.samp_rate = gamp_rate = 400000 307 308 weight_file = open('/home/d/chat4', 'w') 309 weight_file.write("ACK\n") 309 weight_file.close() 311 312 ###############################</pre>					
<pre>303</pre>					
304 # Variables 305 ####################################	302				
<pre>305</pre>					
<pre>306 self.gamp_rate = gamp_rate = 400000 307 308 weight_file = open('/home/c/chat4', 'w') 309 weight_file.write("ACK\n") 310 weight_file.close() 311 312 ###################################</pre>					
<pre>307 308 weight_file = open('/home/c/chat4', 'w') 309 weight_file.write("ACK\n") 310 weight_file.close() 311 312 ###############################</pre>					
308 weight_file = open('/home/a/chat4', 'w') 309 weight_file.write("ACK\n") 310 weight_file.close() 311			-	selr. <u>samp_</u> rate = <u>samp_</u> rate = 400000	
309 weight_file.write("AGK\n") 310 weight_file.close() 311				weight file = $\operatorname{cons}(t/\operatorname{home}(a/\operatorname{chat}(t), t_N))$	
310 weight_file.close() 311					
311 312 • ###################################					
313 # Blocks	312				
	313			# Blooks	

314	
315	<pre>self.uhd_usrp_sink_1 = uhd.usrp_sink(</pre>
316	",".join(("", "")),
317	uhd.stream_args(
318	cpu_format="fo32",
319	channels=range(1),
320),
321)
322	self.uhd_usrp_sink_1.set_samp_rate(samp_rate)
323	<pre>self.uhd_usrp_sink_1.set_time_now(uhd.time_spec(time.time()), uhd.ALL_MBOARDS)</pre>
324	<pre>self.uhd_usrp_sink_1.set_center_freq(1000000000, 0)</pre>
325	<pre>self.uhd_usrp_sink_1.set_gain(40, 0)</pre>
326	<pre>self.uhd_usrp_sink_1.set_antenna('TX/RX', 0)</pre>
327	<pre>self.digital_gmsk_mod_0 = digital.gmsk_mod(</pre>
328	<pre>samples_per_symbol=2,</pre>
329	bt=0.35,
330	verbose=False,
331	log=False,
332	
333	<pre>self.blocks_throttle_0 = blocks.throttle(gr.sizeof_char*1, samp_rate,True)</pre>
334	self.blocks_multiply_const_vxx_1 = blocks.multiply_const_vcc((1,))
335	<pre>self.blocks_head_0 = blocks.head(gr.sizeof_char*1, 100000)</pre>
336	<pre>self.blocks_file_source_0 = blocks.file_source(gr.sizeof_char*1, '/home/c/chat4', True)</pre>
337	self.blocks file source 0.set begin tag(pmt.PMT_NIL)
338	self.blks2_packet_encoder_0 = grc_blks2.packet_mod_b(grc_blks2.packet_encoder(
339	samples per symbol=2,
340	bits per symbol=1,
341	preamble='',
342	access code='',
343	pad_for_usrp=True,
344	
345	payload_length=0,
346	
347	
348	
349	
350 351	# Connections
	# Connections
351 352	
351 352 353	<pre>self.connect((self.blks2_packet_encoder_0, 0), (self.digital_gmsk_mod_0, 0))</pre>
351 352 353 354	<pre>self.connect((self.blks2_packet_encoder_0, 0), (self.digital_gmsk_mod_0, 0)) self.connect((self.blocks_file_source_0, 0), (self.blocks_throttle_0, 0))</pre>
351 352 353 354 355	<pre>self.connect((self.blxs2_packet_encoder_0, 0), (self.digital_gmsk_mod_0, 0)) self.connect((self.blocks_file_source_0, 0), (self.blocks_throttle_0, 0)) self.connect((self.blocks_head_0, 0), (self.blks2_packet_encoder_0, 0))</pre>
351 352 353 354 355 356	<pre>self.connect((self.blks2_packet_encoder_0, 0), (self.digital_gmsk_mod_0, 0)) self.connect((self.blocks_file_source_0, 0), (self.blocks_throttle_0, 0)) self.connect((self.blocks_head_0, 0), (self.blks2_packet_encoder_0, 0)) self.connect((self.blocks_multiply_const_vxx_1, 0), (self.uhd_usrp_sink_1, 0))</pre>
351 352 353 354 355 356 357	<pre>self.connect((self.blks2_packet_encoder_0, 0), (self.digital_gmsk_mod_0, 0)) self.connect((self.blocks_file_source_0, 0), (self.blocks_throttle_0, 0)) self.connect((self.blocks_multiply_const_vxx_1, 0), (self.uhd_usrp_sink_1, 0)) self.connect((self.blocks_throttle_0, 0), (self.blocks_head_0, 0))</pre>
351 352 353 354 355 356	<pre>self.connect((self.blks2_packet_encoder_0, 0), (self.digital_gmsk_mod_0, 0)) self.connect((self.blocks_file_source_0, 0), (self.blocks_throttle_0, 0)) self.connect((self.blocks_head_0, 0), (self.blks2_packet_encoder_0, 0)) self.connect((self.blocks_multiply_const_vxx_1, 0), (self.uhd_usrp_sink_1, 0))</pre>
351 352 353 354 355 356 357 358 359	<pre>self.connect((self.blxs2_packet_encoder_0, 0), (self.digital_gmsk_mod_0, 0)) self.connect((self.blocks_fiele_source_0, 0), (self.blocks_throttle_0, 0)) self.connect((self.blocks_head_0, 0), (self.blks2_packet_encoder_0, 0)) self.connect((self.blocks_multiply_const_vxx_1, 0), (self.uhd_usrp_sink_1, 0)) self.connect((self.blocks_throttle_0, 0), (self.blocks_head_0, 0)) self.connect((self.digital_gmsk_mod_0, 0), (self.blocks_multiply_const_vxx_1, 0))</pre>
351 352 353 354 355 356 357 358 359 360	<pre>####################################</pre>
351 352 353 354 355 356 357 358 359 360 361	<pre>####################################</pre>
351 352 353 354 355 356 357 358 359 360	<pre>####################################</pre>
351 352 353 354 355 356 357 358 359 360 361 362 363	<pre>####################################</pre>
351 352 353 354 355 356 357 358 359 360 361 362 363 364	<pre>####################################</pre>
351 352 353 354 355 356 357 358 359 360 361 362 363	<pre>####################################</pre>
351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366	<pre>####################################</pre>
351 352 353 354 355 356 357 358 359 360 361 362 363 364 365	<pre>####################################</pre>
351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 365 366 367 368	<pre>####################################</pre>
351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 365 366 367 368 369	<pre>####################################</pre>
351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 365 366 366 367 368 369 370	<pre>####################################</pre>
351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 366 365 366 365 366 365 370 371	<pre>####################################</pre>
351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371	<pre>####################################</pre>
351 352 353 354 355 356 357 359 360 361 362 363 364 365 366 365 366 365 366 370 371 372 373	<pre>####################################</pre>
351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 366 370 371 372	<pre>####################################</pre>
351 352 353 354 355 356 357 358 360 361 362 363 363 364 365 366 366 370 371 372 373 374 375	<pre>####################################</pre>
351 352 353 354 355 356 357 358 360 361 362 363 364 365 366 371 371 372 373 374 375 376	<pre>####################################</pre>
351 352 353 354 355 356 357 358 357 358 357 360 361 362 363 364 365 366 366 366 370 371 372 373 374 375 376 377	<pre>####################################</pre>
351 352 353 353 355 356 357 358 360 361 362 366 366 366 366 366 366 367 371 377 377 377 377 377 377	<pre>####################################</pre>
351 352 353 354 355 356 357 388 359 360 361 362 364 365 366 366 366 366 370 371 372 373 374 375 377 377 378 377	<pre>####################################</pre>
351 352 353 354 355 356 357 358 360 361 362 363 363 364 365 366 366 367 366 367 370 371 377 377 377 377 377 377 379 380	<pre>####################################</pre>
351 352 353 354 355 356 357 358 359 360 361 362 363 363 364 365 366 366 370 371 372 377 377 377 377 377 377 377 377 377	<pre>####################################</pre>
351 352 353 354 355 356 356 356 357 358 360 361 362 363 363 363 363 363 363 363 364 365 366 363 371 372 377 377 378 377 378 379 381 382	<pre>####################################</pre>
351 352 353 354 355 355 355 360 361 362 366 366 366 366 366 370 371 373 374 377 377 377 377 377 377 377 379 380 381 379 382 382 382	<pre>####################################</pre>
351 352 353 354 355 356 357 358 359 361 362 363 364 365 364 365 366 366 367 371 373 374 377 377 377 377 377 377 377 377	<pre>####################################</pre>
351 352 353 354 355 356 357 358 359 360 361 362 366 366 366 366 366 370 371 372 373 377 377 377 377 377 377 377 377	<pre>####################################</pre>
351 352 353 354 355 356 357 358 359 361 362 363 364 365 364 365 366 366 367 371 373 374 377 377 377 377 377 377 377 377	<pre>####################################</pre>

(111)	nagel
(Image continued from previous)	pager

388	
389	$g_num = 0$ h_num = 0
390	i num = 0
391	num = 0
392	k_num = 0
393	1_num = 0
394	m_num = 0
395 396	n_num = 0 sum_a = 0
397	sum_b = 0
398	sun_c = 0
399	sum_d = 0
400	sum_e = 0
401	sum_f = 0
402	$a_{\text{min}} = 0$
403 404	sum_h = 0 sum_i = 0
405	sun_i = 0
406	sum k = 0
407	sum 1 = 0
408	sum_m = 0
409	sum_n = 0
410	predict_a = 0
411 412	predict_b = 0 predict_c = 0
413	predict_d = 0
414	predict_e = 0
415	predict_f = 0
416	predict_g = 0
417	predict h = 0
418 419	predict_i = 0 predict_j = 0
420	predict_k = 0
421	predict_l = 0
422	predict_m = 0
423	predict_n = 0
424 425	actual a list = []
425	actual_b_list = [] actual_c_list = []
427	actual d list = []
428	actuallist = []
429	actual_f_list = []
430	actual_g_list = []
431 432	actual h list = []
433	actual_i_list = [] actual_j_list = []
434	actual k list = []
435	actual_1_list = []
436	actual_m_list = []
437	actual_n_list = []
438 439	<pre>predict_a_list = [] predict_b_list = []</pre>
439	predictlist = []
441	predict_list = []
442	predict_e_list = []
443	predict_f_list = []
444	<pre>predict_g_list = [] predict h_list = []</pre>
445 446	<pre>predict _ list = []</pre>
447	predict_list = []
448	predict_k_list = []
449	predict_l_list = []
450	<pre>predict m_list = []</pre>
451 452	predict <u>n list</u> = []
452	while True:
454	weight1 = raw_input("Enter a value for the weight of sensor 1 (integer): ")
455	try:
456	int(weight)
457	if (int(weight) < 0):
458 459	print, "Value cannot be negative. Try again." continue
459	except ValueError:
461	print "Value provided needs to be an integer. Try again."

462 463	continue
464	break
465	while True:
466	weight2 = raw_input("Enter a value for the weight of sensor 2 (integer): ")
467	try:
468	int(weight2)
469	if (int(weight2) < 0):
470 471	print_"Value cannot be negative. Try again." continue
472	except ValueError:
473	print "Value provided needs to be an integer. Try again."
474	continue
475	break
476	while True:
477 478	while irus: weights = raw input ("Enter a value for the weight of sensor 3 (integer): ")
479	try:
480	int (weight3)
481	if (int(weight3) < 0):
482	print, "Value cannot be negative. Try again."
483 484	continue except ValueError:
485	except values root: print, "Value provided needs to be an integer. Try again."
486	continue
487	break
488	
489 490	<pre>while True: weight4 = raw_input("Enter a value for the weight of sensor 4 (integer): ")</pre>
491	weight - ram_input(shier a value for the weight of sensor 4 (integer).) try:
492	int (weight4)
493	if (int(weight4) < 0):
494	print, "Value cannot be negative. Try again."
495 496	continue except ValueError:
496	print_"Value provided needs to be an integer. Try again."
498	continue
499	break
500	while many
501 502	<pre>while True: weight5 = raw_input("Enter a value for the weight of sensor 5 (integer): ")</pre>
503	try:
504	int (weight5)
505	<pre>if (int(weight5) < 0):</pre>
506 507	print, "Value cannot be negative. Try again." continue
508	except ValueError:
509	print "Value provided needs to be an integer. Try again."
510	continue
511	break
512 513	while True:
513	while frue: weights = raw_input("Enter a value for the weight of sensor 6 (integer): ")
515	try:
516	int(weight6)
517	<pre>if (int(weight6) < .0):</pre>
518 519	print _y "Value cannot be negative. Try again." continue
520	except ValueError:
521	print, "Value provided needs to be an integer. Try again."
522	continue
523	break
524 525	while True:
525	while frue: weight7 = raw input("Enter a value for the weight of sensor 7 (integer): ")
527	try:
528	int(weight7)
529	if (int(weight7) < .0):
530 531	print, "Value cannot be negative. Try again."
531	continue except ValueError:
533	print_"Value provided needs to be an integer. Try again."
534	continue
535	break

<pre>S30 weight0 = raw_input("Enter a value for the weight of sensor 0 (integer): ") S30 try: S30 if (int(weight0) < 0): S40 if (int(weight0) < 0): S40 </pre>	536	
<pre>sty : intrespects</pre>	537	while True:
<pre>sty : intrespects</pre>	538	weights = raw input ("Enter a value for the weight of geneor 8 (integer); ")
<pre>set up int(winks) { if (un(vinks) < 0);</pre>		
<pre>status: status: s</pre>	539	try:
<pre>set space spa</pre>	540	int(weight8)
<pre>set space spa</pre>	541	if $(int(weight8) < 0)$:
<pre>style="body:s</pre>		
<pre>set 0 state from the set of the set of</pre>		
<pre>set set set set set set set set set set</pre>	543	continue
<pre>montions track to brack mile run: for functions in (wight) = run; in (wight) = functions in the wight of senser 1 (integer): ") if (integin(") if (integin()) if (inte</pre>	544	except ValueError:
<pre>montions track to brack mile run: for functions in (wight) = run; in (wight) = functions in the wight of senser 1 (integer): ") if (integin(") if (integin()) if (inte</pre>	545	print "Value provided needs to be an integer. Try again."
Image: Sec: Sec: Sec: Sec: Sec: Sec: Sec: Se		
<pre>vils Ture: vils T</pre>		
<pre>viis Trai: viet(viet(This a value for the weight of sensor 0 (integer); ") viet(viet(This a value for the weight of sensor 10 (integer); ") viet(viet(This provide needs to be an integer. Try sgain." continue track viis Trai: viet(viet(This a value for the weight of sensor 10 (integer); ") viet(viet(This a value for the weight of sensor 10 (integer); ") viet(viet(This a value for the weight of sensor 10 (integer); ") viet(viet(This a value for the weight of sensor 11 (integer); ") viet(viet(This a value for the weight of sensor 11 (integer); ") viet(viet(This a value for the weight of sensor 11 (integer); ") viet(viet(This a value for the weight of sensor 11 (integer); ") viet(viet(This a value for the weight of sensor 11 (integer); ") viet(viet(This a value for the weight of sensor 12 (integer); ") viet(viet(This a value for the weight of sensor 12 (integer); ") viet(viet(This a value for the weight of sensor 12 (integer); ") viet(viet(This a value for the weight of sensor 12 (integer); ") viet(viet(This a value for the weight of sensor 12 (integer); ") viet(viet(This a value for the weight of sensor 12 (integer); ") viet(viet(This a value for the weight of sensor 12 (integer); ") viet(viet(This a value for the weight of sensor 12 (integer); ") viet(viet(This a value for the weight of sensor 12 (integer); ") viet(viet(This a value for the weight of sensor 12 (integer); ") viet(viet(This a value for the weight of sensor 12 (integer); ") viet(viet(This a value for the weight of sensor 12 (integer); ") viet(viet(This a value for the weight of sensor 13 (integer); ") viet(viet(This a value for the weight of sensor 13 (integer); ") viet(viet(This a value for the weight of sensor 13 (integer); ") viet(viet(This a value for the weight of sensor 13 (integer); ") viet(viet(This a value for the weight of sensor 13 (integer); ") viet(viet(This a value for the weight of sensor 13 (integer); ") viet(viet(This a value for the weight of sensor 13 (integer); ") viet(viet(This a value for the weight of sensor 13 (integer);</pre>		break
<pre>style="fight: sex_input("finter a value for the weight of sensor 9 (integer): ")</pre>	548	
<pre>style="fight: sex_input("finter a value for the weight of sensor 9 (integer): ")</pre>	549	while True:
<pre>style style s</pre>		weight $9 = raw$ input ("Enter a value for the weight of sensor 9 (integer): ")
<pre>sec pet 'select's 'se</pre>		
<pre>set in the set is a set i</pre>		
<pre>Set 0</pre>	552	int(weight9)
<pre>setup. Sailaber of secure setup. Sailaber of secure setup. Try again." centime break setup. Sailaber of secure setup. Sailaber of secure setup. Try again." centime break setup. Sailaber of secure setup. Sailaber of secure setup. Sailaber of secure 10 (integer): ") try: int(secure secure sec</pre>	553	if (int(weight9) < 0):
<pre>setup. Sailaber of secure setup. Sailaber of secure setup. Try again." centime break setup. Sailaber of secure setup. Sailaber of secure setup. Try again." centime break setup. Sailaber of secure setup. Sailaber of secure setup. Sailaber of secure 10 (integer): ") try: int(secure secure sec</pre>	554	print "Value cannot be negative. Try again."
<pre>secopt ValueTroot: print, "value provided mede to be an integer. Try again." continue weight10 = rw_input("Enter a value for the weight of sensor 10 (integer): ") weight10 = rw_input("Enter a value for the weight of sensor 11 (integer): ") weight10 = rw_input("Enter a value for the weight of sensor 11 (integer): ") weight11 = rw_input("Enter a value for the weight of sensor 11 (integer): ") weight11 = rw_input("Enter a value for the weight of sensor 11 (integer): ") weight11 = rw_input("Enter a value for the weight of sensor 11 (integer): ") try</pre>		
<pre>style="finite-reading-reader by the set of the set</pre>		
<pre>set outlook beak weight 0 = rue: weight 0 = rue: ty; int(reight1) if (int(reight1) < 0): print, "value provided meeds to be an integer. Try again." continue weight 1 = rue: weight 1 = rue: weight 1 = rue: int(reight1) if (int(reight1) < 0): print, "value provided meeds to be an integer. Try again." continue except ValueTror: print(reight1) if (int(reight1) < 0): print, "value provided meeds to be an integer. Try again." continue except ValueTror: print(reight1) if (int(reight1) < 0): print, "value provided meeds to be an integer. Try again." continue except ValueTror: print(reight1) if (int(reight1) < 0): print, "value provided meeds to be an integer. Try again." continue except ValueTror: print(reight1) if (int(reight1) < 0): print, "value provided meeds to be an integer. Try again." continue except ValueTror: print(reight1) if (int(reight1) < 0): print, "value provided meeds to be an integer. Try again." continue except ValueTror: print(reight1) < 0): print, "value provided meeds to be an integer. Try again." continue except ValueTror: print(reight1) < 0): print, "value provided meeds to be an integer. Try again." continue except ValueTror: print(reight1) < 0): print(reight1) < 0): print, "value provided meeds to be an integer. Try again." continue except ValueTror: print(reight1) < 0): print(reight1) <</pre>		
break wills True: try: in (int(wight1)) if	557	print_"Value provided needs to be an integer. Try again."
<pre>while True: weight10 = rew_input("mater a value for the weight of sensor 10 (integer): ") if (int(weight10)): primt, "value senset be negative. Try again." continue break weight11 = rew_input("mater a value for the weight of sensor 11 (integer): ") try: int (weight10)): primt, "value genorided needs to be an integer. Try again." continue except ValueForce: primt, "value genorided needs to be an integer. Try again." continue except ValueForce: primt, "value genorided needs to be an integer. Try again." continue except ValueForce: primt, "value genorided needs to be an integer. Try again." continue except ValueForce: primt, "value genorided needs to be an integer. Try again." continue except ValueForce: primt, "value genorided needs to be an integer. Try again." continue except ValueForce: primt, "value genorided needs to be an integer. Try again." continue except ValueForce: primt, "value genorided needs to be an integer. Try again." continue except ValueForce: primt, "value provided needs to be an integer. Try again." continue except ValueForce: primt, "value provided needs to be an integer. Try again." continue except ValueForce: primt, "value provided needs to be an integer. Try again." continue except ValueForce: primt, "value provided needs to be an integer. Try again." continue except ValueForce: primt, "value provided needs to be an integer. Try again." continue except ValueForce: primt, "value provided needs to be an integer. Try again." continue except ValueForce: primt, "value provided needs to be an integer. Try again." continue except ValueForce: primt, "value provided nee</pre>	558	continue
<pre>while True: weight10 = rew_input("mater a value for the weight of sensor 10 (integer): ") if (int(weight10)): primt, "value senset be negative. Try again." continue break weight11 = rew_input("mater a value for the weight of sensor 11 (integer): ") try: int (weight10)): primt, "value genorided needs to be an integer. Try again." continue except ValueForce: primt, "value genorided needs to be an integer. Try again." continue except ValueForce: primt, "value genorided needs to be an integer. Try again." continue except ValueForce: primt, "value genorided needs to be an integer. Try again." continue except ValueForce: primt, "value genorided needs to be an integer. Try again." continue except ValueForce: primt, "value genorided needs to be an integer. Try again." continue except ValueForce: primt, "value genorided needs to be an integer. Try again." continue except ValueForce: primt, "value genorided needs to be an integer. Try again." continue except ValueForce: primt, "value provided needs to be an integer. Try again." continue except ValueForce: primt, "value provided needs to be an integer. Try again." continue except ValueForce: primt, "value provided needs to be an integer. Try again." continue except ValueForce: primt, "value provided needs to be an integer. Try again." continue except ValueForce: primt, "value provided needs to be an integer. Try again." continue except ValueForce: primt, "value provided needs to be an integer. Try again." continue except ValueForce: primt, "value provided needs to be an integer. Try again." continue except ValueForce: primt, "value provided nee</pre>		break
<pre>84 viib True: 94 viib True: 94 viib True: 95 viib ti * rw_input ("mute a value for the weight of sensor 10 (integer): ") 95 viib viib * rw_input ("mute a value for the weight of sensor 11 (integer): ") 95 viib viib * rw_input ("mute a value for the weight of sensor 11 (integer): ") 95 viib viib * rw/input ("mute a value for the weight of sensor 12 (integer): ") 95 viib viib * rw/input ("mute a value for the weight of sensor 12 (integer): ") 95 viib viib * rw/input ("mute a value for the weight of sensor 12 (integer): ") 95 viib viib * rw/input ("mute a value for the weight of sensor 12 (integer): ") 95 viib viib * rw/input ("mute a value for the weight of sensor 12 (integer): ") 95 viib viib * rw/input ("mute a value for the weight of sensor 12 (integer): ") 95 viib viib * rw/input ("mute a value for the weight of sensor 12 (integer): ") 95 viib * rw/input ("muter a value for the weight of sensor 12 (integer): ") 95 viib * rw/input ("muter a value for the weight of sensor 12 (integer): ") 95 viib * rw/input ("muter a value for the weight of sensor 12 (integer): ") 95 viib * rw/input ("muter a value for the weight of sensor 12 (integer): ") 95 viib * rw/input ("muter a value for the weight of sensor 12 (integer): ") 95 viib * rw/input ("muter a value for the weight of sensor 12 (integer): ") 95 viib * rw/input ("muter a value for the weight of sensor 12 (integer): ") 95 viib * rw/input ("muter a value for the weight of sensor 12 (integer): ") 95 viib * rw/input ("muter a value for the weight of sensor 13 (integer): ") 95 viib * rw/input ("muter a value for the weight of sensor 13 (integer): ") 95 viib * rw/input ("muter a value for the weight of sensor 13 (integer): ") 95 viib * rw/input ("muter a value for the weight of sensor 13 (integer): ") 95 viib * rw/input ("muter a value for the weight of sensor 13 (integer): ") 95 viib * rw/input ("muter a value for the weight of sensor 13 (integer): ") 95 viib * rw/input ("muter a value for the weight of sensor 13 (integer): ") 95 viib</pre>		
<pre>set if if</pre>		
<pre>set of the set of</pre>	561	while True:
<pre>set state (veright10) / if (int(veright10) / s):</pre>	562	<pre>weight10 = raw_input("Enter a value for the weight of sensor 10 (integer): ")</pre>
<pre>set state (veright10) / if (int(veright10) / s):</pre>	563	trv:
<pre>set if (int(seight)) < 0):</pre>		
<pre>sec print "Value cannot be negative. Try again." continue except ValueTrror: print "Value provided needs to be an integer. Try again." continue except ValueTrror: print (veight1) frint(veight1) fr</pre>		
<pre>secept Vile:Tron:</pre>	565	if (int(weight10) < 0):
<pre>second ValueTron:: print_"Value provided needs to be an integer. Try again." ontinue break value for the weight of sensor 11 (integer): ") try: if (int (weight11) ff (int (weight12) < 0): trint_"Value provided needs to be an integer. Try again." continue rest value True: value for the weight of sensor 12 (integer): ") try: int(weight12) < 0): print_"Value provided needs to be an integer. Try again." continue rest value for the weight of sensor 12 (integer): ") try: int(weight12) < 0): print_"Value provided needs to be an integer. Try again." continue rest value for the weight of sensor 12 (integer): ") try: int(weight12) < 0): print_"Value provided needs to be an integer. Try again." continue break value for the weight of sensor 13 (integer): ") try: int(weight13) < 0): print_"Value provided needs to be an integer. Try again." continue break value for the weight of sensor 13 (integer): ") try: int(weight13) < 0): print_"Value provided needs to be an integer. Try again." continue break value for the weight of sensor 13 (integer): ") try: if (int(weight13) < 0): print_"Value provided needs to be an integer. Try again." continue break value for the weight of sensor 13 (integer): ") try: if (int(weight13) < 0): print_"Value provided needs to be an integer. Try again." continue break continue break value for the weight of sensor 13 (integer): ") try: if (int(weight13) < 0): print_Value for the weight of sensor 13 (integer): ") try: if (int(weight13) < 0): print_Value for the weight of sensor 13 (integer): ") try: for (integer): ") try: for (integer): ") try: for (intege</pre>	566	print "Value cannot be negative. Try again."
<pre>second ValueTron:: print_"Value provided needs to be an integer. Try again." ontinue break value for the weight of sensor 11 (integer): ") try: if (int (weight11) ff (int (weight12) < 0): trint_"Value provided needs to be an integer. Try again." continue rest value True: value for the weight of sensor 12 (integer): ") try: int(weight12) < 0): print_"Value provided needs to be an integer. Try again." continue rest value for the weight of sensor 12 (integer): ") try: int(weight12) < 0): print_"Value provided needs to be an integer. Try again." continue rest value for the weight of sensor 12 (integer): ") try: int(weight12) < 0): print_"Value provided needs to be an integer. Try again." continue break value for the weight of sensor 13 (integer): ") try: int(weight13) < 0): print_"Value provided needs to be an integer. Try again." continue break value for the weight of sensor 13 (integer): ") try: int(weight13) < 0): print_"Value provided needs to be an integer. Try again." continue break value for the weight of sensor 13 (integer): ") try: if (int(weight13) < 0): print_"Value provided needs to be an integer. Try again." continue break value for the weight of sensor 13 (integer): ") try: if (int(weight13) < 0): print_"Value provided needs to be an integer. Try again." continue break continue break value for the weight of sensor 13 (integer): ") try: if (int(weight13) < 0): print_Value for the weight of sensor 13 (integer): ") try: if (int(weight13) < 0): print_Value for the weight of sensor 13 (integer): ") try: for (integer): ") try: for (integer): ") try: for (intege</pre>	567	continue
<pre>second the frue:</pre>		
<pre>prove the second of the s</pre>		
1 0 brak 0 while True: weightli = rww_input("Enter a value for the weight of sensor 11 (integer): ") 0 try: int(weightli) if (int(weightli) if (
<pre>value value True: veight1 = raw_input("Enter a value for the weight of sensor 11 (integer): ") try; int(weight1)) if (int(weight1)) < 0): print_"value cannot be negative. Try again." continue except ValueTror: veight12 = raw_input("Enter a value for the weight of sensor 12 (integer): ") try: if (int(weight12), < 0): print_"value cannot be negative. Try again." continue setopt ValueTror: veight12 = raw_input("Enter a value for the weight of sensor 12 (integer): ") try: if (int(weight12), < 0): print_"value cannot be negative. Try again." continue setopt ValueTror: veight13 = raw_input("Enter a value for the weight of sensor 13 (integer): ") try: if (int(weight12), < 0): print_"value cannot be negative. Try again." continue break veight13 = raw_input("Enter a value for the weight of sensor 13 (integer): ") try: if (weight13) if (int(weight13), < 0): print_"value cannot be negative. Try again." continue except ValueTror: print_"value provided needs to be an integer. Try again." continue break veight13 = raw_input("Enter a value for the weight of sensor 13 (integer): ") try: if (int(weight13) if (int(weight13), < 0): print_"value provided needs to be an integer. Try again." continue except ValueTror: print_"value provided needs to be an integer. Try again." continue except ValueTror: print_"value provided needs to be an integer. Try again." continue except ValueTror: print_"value provided needs to be an integer. Try again." continue break except ValueTror: print_"value provided needs to be an integer. Try again." continue break except ValueTror: print_"value provided needs to be an integer. Try again." continue break except ValueTror: print_"value provided needs to be an integer. Try again." continue break except ValueTror: print_"value provided needs to be an integer. Try again." continue break except ValueTror: print_"value provided needs to be an integer. Try again." continue break except ValueTror: provided needs to be an integer. Try</pre>	570	continue
<pre>513 0 while True: weight1] = raw_input("Enter a value for the weight of sensor 11 (integer): ") try: int(weight1]) if (int(weight1]) < 0): print, "Value cannot be negative. Try again." continue break while True: weight12 = raw_input("Enter a value for the weight of sensor 12 (integer): ") try: int(weight12) < 0): print, "Value cannot be negative. Try again." continue setSet 0 while True: weight12 = raw_input("Enter a value for the weight of sensor 12 (integer): ") try: int(weight12) < 0): print, "Value cannot be negative. Try again." continue setSet 0 while True: weight13 = raw_input("Enter a value for the weight of sensor 12 (integer): ") try: int(weight13) < 0): print, "Value cannot be negative. Try again." continue weight13 = raw_input("Enter a value for the weight of sensor 13 (integer): ") try: int(weight13) < 0): print, "Value cannot be negative. Try again." continue setSet 0 weight13 = raw_input("Enter a value for the weight of sensor 13 (integer): ") try: int(weight13) = fraw_input("Enter a value for the weight of sensor 13 (integer): ") try: int(weight13) = fraw_input("Enter a value for the weight of sensor 13 (integer): ") try: int(weight13) = fraw_input("Enter a value for the weight of sensor 13 (integer): ") try: int(weight13) = fraw_input("Enter a value for the weight of sensor 13 (integer): ") try: int(weight13) = fraw_input("Enter a value for the weight of sensor 13 (integer): ") if (int (weight13) = fraw_input ("Enter a value for the weight of sensor 13 (integer): ") if (int (weight13) = fraw_input ("Enter a value for the weight of sensor 13 (integer): ") if (int (weight13) = fraw_input ("Enter a value for the weight of sensor 13 (integer): ") if (int (weight13) = fraw_input ("Enter a value for the weight of sensor 13 (integer): ") if (int (weight13) = fraw_input ("Enter a value for the weight of sensor 13 (integer): ") if (int (weight13) = fraw_input ("Enter a value for the weight of sensor 13 (integer): ") if (int (weight13) = fraw_input ("E</pre>	571	break
<pre>Set veight1 = raw_input("Enter a value for the weight of sensor 11 (integer): ") try: int(weight1) < 0): primt,"value cannot be negative. Try again." continue except ValueFror: primt,"value cannot be negative. Try again." continue weight12 = raw_input("Enter a value for the weight of sensor 12 (integer): ") try: int(weight12) int(weight12) < 0): primt,"value cannot be negative. Try again." continue weight12 = raw_input("Enter a value for the weight of sensor 12 (integer): ") try:</pre>	572	
<pre>Set veight1 = raw_input("Enter a value for the weight of sensor 11 (integer): ") try: int(weight1) < 0): primt,"value cannot be negative. Try again." continue except ValueFror: primt,"value cannot be negative. Try again." continue weight12 = raw_input("Enter a value for the weight of sensor 12 (integer): ") try: int(weight12) int(weight12) < 0): primt,"value cannot be negative. Try again." continue weight12 = raw_input("Enter a value for the weight of sensor 12 (integer): ") try:</pre>	573	while True.
55 try: 56 int(weight1) < 0):		
<pre>int (weight1) it (weight2) < 0:</pre>	574	weightil = raw_input("Enter a value for the weight of sensor 11 (integer): ")
<pre>if (int(weight11) < 0): print_"Value cannot be negative. Try again." oontinue eccept ValueTror: print_"Value provided needs to be an integer. Try again." continue break vhile True: weight12 = raw_input("Enter a value for the weight of sensor 12 (integer): ") try: int(weight12) < 0): int(weight12) < 0): int(weight12) < 0): if (int(weight12) < 0): if (int(weight12) < 0):</pre>	575	try:
<pre>if (int(weight11) < 0): print_"Value cannot be negative. Try again." oontinue eccept ValueTror: print_"Value provided needs to be an integer. Try again." continue break vhile True: weight12 = raw_input("Enter a value for the weight of sensor 12 (integer): ") try: int(weight12) < 0): int(weight12) < 0): int(weight12) < 0): if (int(weight12) < 0): if (int(weight12) < 0):</pre>	576	int(weight11)
<pre>579 print_"Value cannot be negative. Try again." 579 print_"Value provided needs to be an integer. Try again." 580 print_"Value provided needs to be an integer. Try again." 580 while True: 580 weight12 = raw_input("Enter a value for the weight of sensor 12 (integer): ") 587 try: 588 if (int(weight12)) { 589 print_"Value cannot be negative. Try again." 589 print_"Value provided needs to be an integer. Try again." 589 while True: 580 weight13 = raw_input("Enter a value for the weight of sensor 12 (integer): ") 581 print_"Value cannot be negative. Try again." 580 weight13 = raw_input("Enter a value for the weight of sensor 13 (integer): ") 580 print_"Value provided needs to be an integer. Try again." 580 weight13 = raw_input("Enter a value for the weight of sensor 13 (integer): ") 580 try: 580 try: 580 try: 580 ontinue 580 print_"Value provided needs to be an integer. Try again." 580 ontinue 580 print_Value for the weight of sensor 13 (integer): ") 580 try: 580 ontinue 580 print_Value provided needs to be an integer. Try again." 580 ontinue 580 print_Value provided needs to be an integer. Try again." 580 ontinue 580 print_Value provided needs to be an integer. Try again." 580 ontinue 580 ontin</pre>	577	
599 oontime 800 except ValueError: 810 print, "Value provided needs to be an integer. Try again." 821 break 832 int(reight12) 833 int(reight12) 834 int(reight12) 835 int(reight12) 836 int(reight12) 837 int(reight12) 838 int(reight12) 839 int(reight12) 839 int(reight12) 839 int(reight12) 839 int(reight12) 839 int(reight12) 839 int(reight12) 830 int(reight12) 831 int(reight12) 832 except ValueError: 833 int(reight13) 834 int(reight13) 835 int(reight13) 836 int(reight13) 837 int(reight13) 838 int(reight13) 849 int(reight13) 840 int(reight13) 841 int(reight13) 842 int(reight1		
500 • except ValueTror: 511 print, "Value provided needs to be an integer. Try again." 523 • break 534 • wille True: 535 • try: 536 • try: 537 • try: 538 • try: 539 • try: 531 • try: 532 • try: 533 • try: 534 • try: 535 • try: 536 • try: 537 • try: 538 • try: 539 • try: 531 • try: 533 • try: 534 • try: 535 • try: 536 • try: 537 • try: 538 • try: 539 • try: 531 • try: 533 • try: 534 • try: 535 • try: 536 • try: 537 • try: 538 • try: 539 • try: 531 • try: 532 • try: 533 • try: 534 • try: 535 • try: 536 • try: 537 • try: 538 • try: 539 • try: 531 • try: 532 • try: 533 • try: 534 • try: 535 • try: 536 • try: 53		
Sile print_"Value provided needs to be an integer. Try again." Sile break Sile int(weight12 = raw_input("Enter a value for the weight of sensor 12 (integer): ") Sile int(weight12) Sile int(weight12) Sile print_"Value cannot be negative. Try again." Sile int(weight12) Sile print_"Value cannot be negative. Try again." Sile int(weight13) <	579	continue
<pre>sec ontinue sec ontinue s</pre>	580	except ValueError:
<pre>sec ontinue sec ontinue s</pre>	581	print "Value provided needs to be an integer. Try again."
<pre>bis break bis break bis break bis break bis bis bis bis break bis bis bis bis bis bis bis bis bis bis</pre>	582	
<pre>set set set set set set set set set set</pre>		
<pre>ses veight12 = raw_input("Enter a value for the weight of sensor 12 (integer): ") try: int(weight12) if (int(weight12)</pre>		Dreak
<pre>sec veight12 = raw_input ("Enter a value for the weight of sensor 12 (integer): ") try: int(weight12) < 0): print, "Value cannot be negative. Try again." ontinue except ValueFrror: print, "Value provided needs to be an integer. Try again." ontinue while True: weight13 = raw_input ("Enter a value for the weight of sensor 13 (integer): ") try: int (weight13) if (int (weight13) < 0): print, "Value cannot be negative. Try again." continue continue int (weight13) if (int (weight13) < 0): print, "Value cannot be negative. Try again." continue continue weight13 = raw_input ("Enter a value for the weight of sensor 13 (integer): ") try: continue con</pre>		
<pre>set try: int (weight12) <_0):</pre>	585	while True:
<pre>set try: int (weight12) <_0):</pre>	586	weight12 = raw input("Enter a value for the weight of sensor 12 (integer): ")
<pre>sss int(weight12) sss if (int(weight12) < 0): sss if (int(weight12) < 0): print_"Value cannot be negative. Try again." continue ssc print_"Value provided needs to be an integer. Try again." continue sss continue sss </pre>	587	
<pre>if (int(weight12) < 0): print_"Value cannot be negative. Try again." continue except ValueError: print_"Value provided needs to be an integer. Try again." continue break ///////////////////////////////////</pre>		
590 image: print_"Value common be negative. Try again." 591 continue 592 except ValueError: 593 oontinue 594 continue 595 break 596 break 597 while True: 598 weight13 = raw_input("Enter a value for the weight of sensor 13 (integer): ") 599 try: 591 int(weight13) 601 int(weight13) 602 print_"Value cannot be negative. Try again." 603 except ValueError: 504 print_"Value provided needs to be an integer. Try again." 605 print_"Value provided needs to be an integer. Try again." 606 continue 607 break		
591 continue 592 continue 593 print_"Value provided needs to be an integer. Try again." 594 continue 595 break 596 while True: 598 weight13 = raw_input("Enter a value for the weight of sensor 13 (integer): ") 599 try: 599 int(weight13) 601 int(weight13) 602 print_"Value cannot be negative. Try again." 603 continue 604 except ValueError: 605 print_"Value provided needs to be an integer. Try again." 606 continue 607 break		
<pre>sec except ValueError: print "Value provided needs to be an integer. Try again." continue set set set set set set set se</pre>	590	print, "Value cannot be negative. Try again."
<pre>sec except ValueError: print "Value provided needs to be an integer. Try again." continue set set set set set set set se</pre>	591	continue
<pre>593 print_"Value provided needs to be an integer. Try again." 594 continue 595 break 597 while True: 598 weight13 = raw_input("Enter a value for the weight of sensor 13 (integer): ") 599 try: 600 int(weight13) 601 if (int(weight13) < 0): 602 print_"Value cannot be negative. Try again." 603 print_"Value cannot be negative. Try again." 604 except ValueError: 605 print_"Value provided needs to be an integer. Try again." 606 continue 607 break 608</pre>	592	except ValueError:
594 continue 595 break 596 while True: 597 while True: 598 int(weight13 = raw_input("Enter a value for the weight of sensor 13 (integer): ") 599 try: 600 int(weight13) 601 if (int(weight13)) 602 print_"Value cannot be negative. Try again." 603 continue 604 except ValueError: 605 print_"Value provided needs to be an integer. Try again." 606 continue 607 break		
595 break 596 while True: 597 weight13 = raw_input ("Enter a value for the weight of sensor 13 (integer): ") 598 try: 600 int (weight13) 601 int (weight13) 602 ontinue 603 continue 604 except ValueError: 605 print_"Value provided needs to be an integer. Try again." 606 break 608 break		
<pre>ss6 ss7 ss8 ss8 ss8 ss8 weight13 = raw_input("Enter a value for the weight of sensor 13 (integer): ") ss8 ss8 ss8 ss8 ss8 ss8 ss8 ss8 ss8 ss</pre>		
597 while True: 588 weight13 = raw_input("Enter a value for the weight of sensor 13 (integer): ") 599 try: 600 int(weight13) 601 if (int(weight13)) 602 print,"Value cannot be negative. Try again." 603 except ValueError: 604 except ValueError: 605 oontinue 606 oontinue 607 break	595	break
<pre>sea weight13 = raw_input("Enter a value for the weight of sensor 13 (integer): ") sea try: if (int(weight13) if (int(weight13) if (int(weight13) < 0): print_"Value cannot be negative. Try again." continue conti</pre>	596	
<pre>sea weight13 = raw_input("Enter a value for the weight of sensor 13 (integer): ") sea try: if (int(weight13) if (int(weight13) if (int(weight13) < 0): print_"Value cannot be negative. Try again." continue conti</pre>	597	while True:
599 try: 600 int(weight13) 601 if(int(weight13) < 0):		
600 int (weight13) 601 if (int (weight13) < 0):		
601 if (int(weight13) < 0):		
601 if (int(weight13) < 0):	600	int (weight13)
602 print_"Value cannot be negative. Try again." 603 continue 604 cexcept ValueError: 605 print_"Value provided needs to be an integer. Try again." 606 continue 607 break	601	
603 continue 604 continue 605 print_"Value provided needs to be an integer. Try again." 606 continue 607 break		
604 except ValueError: 605 print_"Value provided needs to be an integer. Try again." 606 continue 607 break		
605 print_"Value provided needs to be an integer. Try again." 606 continue 607 break		
606 continue 607 break 608	604	
606 continue 607 break 608	605	print "Value provided needs to be an integer. Try again."
607 break 608	606	
608	607	
609 😓 while True:		
	609	while True:

	610	weight14 = raw_input("Enter a value for the weight of sensor 14 (integer): ")
a a	611	try:
64 0 pottime 65 0 pottime 66 0 pottime 67 0 pottime 68 0 pottime 68 0 pottime 68 0 pottime 68 velpt_ile pottime 68 ile pottime 68 ile pottile <t< th=""><th></th><th></th></t<>		
1		
a prompt function a b provide to be an integer. Try epin." a b wigt_file*controlweight *) a wigt_file*controlweight *) b a wigt_file*controlweight *)		
0 vertime vertime 0 vertime vertime 000000000000000000000000000000000000	616	
0 brak 0 wight_[lit_scipil_(pace_[data], **)] 0 wight_[lit_scipit_(pace_[data], **]) 0 wight_[lit_scipit_(pace_[data], **]) 0 wight_[lit_scipit_[data], **]) 0 wight_[lit_scipit_[data], **]) 0 theratio wight_[lit_scipit_[data], **]) 0 wight_[lit_scipit_[data], **]) 0 wight_[lit_scipit_[data], **]) 0 wight_[lit_scipit_[data], **]) 0 wight_[lit_scipit_[data], *] <	617	print, "Value provided needs to be an integer. Try again."
<pre>setup:_file = open('/back/shalf, 'w') setup:_file = open('/back/shalf, 'w') setup:_file = vise(back + '.) setup:_file = v</pre>		break
<pre>Set Set Set Set Set Set Set Set Set Set</pre>		
<pre>set set set set set set set set set set</pre>		weight_life - open('/nome/o/chats', 'W')
<pre>def v vsigh_file.vrig(vsight) * .') def vsight_file.vrig(vsight) * .') def vsight</pre>		weight file.write(weight1 + ',')
<pre>64 vipt</pre>		
<pre>249 wight_file.vic(wight) +,') 249 wight_file.vic(wight) +,') 249 wight_file.vic(wight) +,') 249 wight_file.vic(wight) +,') 240 wight_file.vic(wight) +,') 241 wight_file.vic(wight) +,') 242 wight_file.vic(wight) +,') 243 wight_file.vic(wight) +,') 244 wight_file.vic(wight) +,') 245 wight_file.vic(wight) +,') 246 wight_file.vic(wight) +,') 247 wight_file.vic(wight) +,') 248 wight_file.vic(wight) +,') 249 wight_file.vic(wight)</pre>	625	weight_file.write(weight3 + ',')
<pre>dig</pre>		
<pre>Geg vsight_file.vic(vsight? +.') Gig vsight_file.vic(vsight?</pre>		
<pre>set set set set set set set set set set</pre>		
<pre>setset_file_rise (respect = ',') setset_file_rise (respect = ', ', ', ',') setset_file_rise (respect = ', ', ', ', ',') setset_file_rise (respect = ', ', ', ', ', ', ', ', ', ', ', ', ',</pre>		
<pre>segnt_file_write(weight) *) segnt_file_write(weight) *) segnt_file_write(weight)</pre>		
<pre>si</pre>	632	
<pre>signer_file.wrise(weight14 + ',')</pre>		
<pre>set vsipt_file.write(wsipt:4 + ','n') vsipt_file.vrite(wsipt:4 + ','n') vsipt_file.vrite() vsipt_file.vite() vsipt_file.vrite()</pre>		
<pre>wight_file.close() wight_file.close() th = chat() th = chat() th = chat() th = cont() th = co</pre>		
<pre>set vsint_fl.close()</pre>		weight_lile.write(Weightia + ', /n')
<pre>i</pre>		weight file.close()
<pre>file tb.stst() tb.sts() tb.sts(</pre>		
<pre>444 time.step(1) 455 456 457 458 458 458 458 458 458 458 458</pre>	640	tb = chat()
<pre>443 tb.stop() 444 455 tb.stop() 456 final.close() 457 458 final.close() 459 459 final.close() 459 459 final.close() 459 459 final.close() 450 fina</pre>		
<pre>final = open('/home/o/final', 'w') final.close() final.close() final.close() change0_list = 0 do d</pre>		
<pre>fin1 = open('/hom/o//fin1', 'w') fin1.close() fin1.close() fin1.close() packet_tot = 0 fin1.close() changel_list = [] changel_list = []</pre>		tp.stop()
<pre>final.close()</pre>		final = open('/home/g/final', 'w')
<pre>69</pre>		
<pre>699</pre>	647	
<pre>60</pre>		packet_tot = 0
<pre>61 change1_list = [] 62 change2_list = [] 63 change4_list = [] 64 change4_list = [] 65 change6_list = [] 66 change6_list = [] 67 change1_list = [] 68 change6_list = [] 68 change6_list = [] 68 change1_list = [] 69 change1_list = [] 60 change1_list = [] 61 change1_list = [] 62 change1_list = [] 63 change1_list = [] 64 change1_list = [] 65 change2_list = [] 66 change1_list = [] 67 change1_list = [] 68 change2_list = [] 69 change2_list = [] 60 change1_list = [] 61 change1_list = [] 62 change2_list = [] 63 change2_list = [] 64 change1_list = [] 65 change2_list = [] 66 change1_list = [] 67 change1_list = [] 68 change2_list = [] 68 change2_list = [] 69 change2_list = [] 60 change1_list = [] 60 change1_list = [] 61 change1_list = [] 62 change2_list = [] 63 change2_list = [] 64 change1_list = [] 65 change2_list = [] 65 change2_list = [] 66 change1_list = [] 66 change1_list = [] 67 change1_list = [] 68 change2_list = [] 68 c</pre>		change list = 1
<pre>622 change2_list = [] 633 change4_list = [] 634 change4_list = [] 635 change5_list = [] 636 change6_list = [] 637 change7_list = [] 638 change6_list = [] 639 change6_list = [] 640 change1_list = [] 640 change1_list = [] 641 change1_list = [] 642 change1_list = [] 643 sum_change1 = 0 644 sum_change1 = 0 645 sum_change3 = 0 646 sum_change3 = 0 647 sum_change5 = 0 740 sum_change6 = 0 751 sum_change6 = 0 753 sum_change6 = 0 754 sum_change1 = 0 755 sum_change1 = 0 755 sum_change1 = 0 756 sum_change1 = 0 757 sum_change1 = 0 757 sum_change1 = 0 758 sum_change1 = 0 759 sum_change1 = 0 759 sum_change1 = 0 750 sum_change1 =</pre>	651	
<pre>644 change4_list = [] 655 change5_list = [] 666 change5_list = [] 667 change9_list = [] 668 change9_list = [] 669 change10_list = [] 660 change10_list = [] 660 change11ist = [] 661 change11_list = [] 662 change20_list = [] 663 change30_list = [] 664 sum_change0 = 0 665 sum_change0 = 0 666 sum_change1 = 0 666 sum_change5 = 0 667 sum_change5 = 0 668 sum_change5 = 0 670 sum_change5 = 0 671 sum_change5 = 0 673 sum_change5 = 0 674 sum_change5 = 0 675 sum_change1 = 0 675 sum_change1 = 0 676 sum_change1 = 0 677 sum_change1 = 0 678 sum_change1 = 0 679 vhile True: 680 tb = text_rx() 681 tb = text_r() 681 tb = text_r() 681 tb = text_r() 682 tb = text_r() 684 tb = text_r() 685 tb = text_r() 68</pre>	652	
<pre>655 change5_list = [] 666 change6_list = [] 667 change6_list = [] 668 change9_list = [] 668 change0_list = [] 669 change0_list = [] 660 change1_list = [] 661 change11_list = [] 662 change12_list = [] 663 sum_change0 = 0 664 sum_change0 = 0 665 sum_change0 = 0 666 sum_change0 = 0 666 sum_change0 = 0 667 sum_change0 = 0 668 sum_change0 = 0 669 sum_change0 = 0 669 sum_change0 = 0 669 sum_change0 = 0 670 sum_change0 = 0 671 sum_change0 = 0 672 sum_change0 = 0 673 sum_change0 = 0 674 sum_change1 = 0 675 sum_change1 = 0 676 sum_change1 = 0 677 sum_change1 = 0 678 sum_change1 = 0 679 vhile True: 680 tb = text_rx() 681 tb = text_rx() 6</pre>	653	change3_list = []
<pre>656 change_list = [] 657 change_list = [] 659 change_list = [] 659 change_list = [] 659 change_list = [] 660 changel_list = [] 661 changel_list = [] 662 changel_list = [] 663 changel = 0 664 sum_change 0 = 0 665 sum_change 2 = 0 666 sum_change 2 = 0 667 sum_change 2 = 0 668 sum_change 3 = 0 669 sum_change 4 = 0 669 sum_change 5 = 0 670 sum_change 6 = 0 671 sum_change 7 = 0 672 sum_change 8 = 0 673 sum_change 1 = 0 674 sum_change 1 = 0 675 sum_change 1 = 0 676 sum_change 1 = 0 677 sum_change 1 = 0 678 sum_change 1 = 0 679 vhile True: 680 tb = text_rx() 681 tb = text_rx() 6</pre>	654	
<pre>657 change7_list = [] change8_list = [] 659 change9_list = [] 660 change1_list = [] 661 change1_list = [] 662 change1_list = [] 663 change1_list = [] 664 sum_change1 = 0 665 sum_change2 = 0 666 sum_change2 = 0 667 sum_change4 = 0 668 sum_change4 = 0 669 sum_change5 = 0 670 sum_change6 = 0 671 sum_change7 = 0 671 sum_change7 = 0 672 sum_change1 = 0 673 sum_change1 = 0 674 sum_change1 = 0 675 sum_change1 = 0 676 sum_change1 = 0 676 sum_change1 = 0 677 sum_change1 = 0 678 sum_change1 = 0 679 while True: 680 tb = text_x() 681 tb = text_t() 681 t</pre>		
<pre>659 change8_list = [] 559 change9_list = [] 560 change0_list = [] 561 change1_list = [] 562 change1_list = [] 563 change1_list = [] 564 sum_change0 = 0 565 sum_change0 = 0 566 sum_change2 = 0 567 sum_change3 = 0 568 sum_change6 = 0 570 sum_change6 = 0 571 sum_change6 = 0 571 sum_change6 = 0 573 sum_change6 = 0 573 sum_change0 = 0 573 sum_change0 = 0 574 sum_change0 = 0 575 sum_change1 = 0 576 sum_change1 = 0 577 sum_change1 = 0 578 sum_change1 = 0 579 while True: 579 the text_x() 581 the start() 582 the text_start() 585 the start() 585 the start()</pre>		
<pre>659 change9_list = [] 660 change1_list = [] 660 change1_list = [] 661 change1_list = [] 662 change1_list = [] 663 change1_list = [] 664 sum_change0 = 0 665 sum_change1 = 0 666 sum_change3 = 0 668 sum_change3 = 0 669 sum_change5 = 0 669 sum_change6 = 0 671 sum_change6 = 0 671 sum_change0 = 0 672 sum_change0 = 0 673 sum_change0 = 0 674 sum_change1 = 0 675 sum_change1 = 0 675 sum_change1 = 0 676 sum_change1 = 0 677 sum_change1 = 0 678 the left sum_change1 = 0 679 the left sum_change1 = 0 679 the left sum_change1 = 0 670 the left sum_change1 = 0 671 the sum_change1 = 0 672 the left sum_change1 = 0 673 the left sum_change1 = 0 674 the left sum_change1 = 0 675 the left sum_change1 = 0 676 the left sum_change1 = 0 677 the left sum_change1 = 0 678 the left sum_change1 = 0 679 the left sum_change1 = 0 670 the left sum_ch</pre>	658	
<pre>660 changel0_list = [] 661 changel1_list = [] 662 changel2_list = [] 663 changel3_list = [] 664 sum_change0 = 0 665 sum_change2 = 0 666 sum_change2 = 0 667 sum_change3 = 0 668 sum_change4 = 0 669 sum_change5 = 0 70 sum_change6 = 0 671 sum_change6 = 0 673 sum_change8 = 0 673 sum_change1 = 0 674 sum_change1 = 0 675 sum_change1 = 0 676 sum_change1 = 0 677 sum_change1 = 0 678 sum_change1 = 0 679 while True: 679 tb = text_rx() 681 tb.start() 682 tb.start()</pre>	659	
<pre>662 changel2_list = [] 663 changel2_list = [] 664 sum_change0 = 0 665 sum_change1 = 0 666 sum_change3 = 0 667 sum_change4 = 0 668 sum_change5 = 0 669 sum_change5 = 0 670 sum_change6 = 0 671 sum_change7 = 0 672 sum_change8 = 0 673 sum_change1 = 0 674 sum_change1 = 0 675 sum_change1 = 0 676 sum_change1 = 0 677 sum_change1 = 0 678 sum_change1 = 0 679 vmlle True: 600 tb = test_tx() 601 tb.start() 601 datal = open('/home/o/test_file2', 'r+') </pre>	660	changel0_list = []
<pre>663 change1_list = [] 664 sum_change0 = 0 665 sum_change1 = 0 666 sum_change2 = 0 667 sum_change3 = 0 668 sum_change4 = 0 669 sum_change5 = 0 670 sum_change5 = 0 671 sum_change6 = 0 671 sum_change8 = 0 673 sum_change9 = 0 674 sum_change1 = 0 675 sum_change1 = 0 676 sum_change1 = 0 677 sum_change1 = 0 678 sum_change1 = 0 679 while True: 679 tb = text_rx() 681 tb = text_rx() 681 tb = text_r() 682 tb = text_r() 681 tb = text_r() 682 tb = text_r() 684 tb =</pre>	661	
664 sum_change0 = 0 665 sum_change1 = 0 666 sum_change3 = 0 667 sum_change4 = 0 668 sum_change5 = 0 669 sum_change6 = 0 671 sum_change6 = 0 672 sum_change6 = 0 673 sum_change10 = 0 674 sum_change11 = 0 675 sum_change12 = 0 676 sum_change13 = 0 677 sum_change14 = 0 678 sum_change10 = 0 679 while True: 670 sum_change13 = 0 671 sum_change13 = 0 672 sum_change13 = 0 673 sum_change13 = 0 674 start(n) 675 sum_change13 = 0 676 start(n) 677 start(n) 680 tb = text_rx(n) tb = text_rx(n) tb.start(n) 681 tb.start(n) 682 data1 = open('home/a/test_file2', 'r+')	662	
<pre>sum_change1 = 0 sum_change2 = 0 sum_change3 = 0 sum_change3 = 0 sum_change4 = 0 sum_change5 = 0 sum_change6 = 0 sum_change7 = 0 sum_change7 = 0 sum_change8 = 0 sum_change1 = 0 sum_change10 = 0 sum_change11 = 0 sum_change11 = 0 sum_change12 = 0 sum_change13 = 0 tb = text_rx() tb = text_rx() tb = text_r() data1 = open('/home/a/test_file2', 'r+')</pre>		
<pre>666 sum_change2 = 0 667 sum_change3 = 0 668 sum_change4 = 0 669 sum_change5 = 0 669 sum_change6 = 0 671 sum_change6 = 0 672 sum_change8 = 0 673 sum_change8 = 0 674 sum_change1 = 0 675 sum_change11 = 0 676 sum_change12 = 0 677 sum_change13 = 0 679 while True: 680 tb = text_rx() 681 tb.start() 682 db = copn('/home/a/test_file2', 'r+')</pre>	665	
668 sum_change4 = 0 669 sum_change5 = 0 670 sum_change6 = 0 671 sum_change7 = 0 672 sum_change8 = 0 673 sum_change10 = 0 674 sum_change11 = 0 675 sum_change12 = 0 676 sum_change13 = 0 677 sum_change12 = 0 678 sum_change12 = 0 679 while True: 680 tb = text_rx() tb = text_rx() tb.start() 681 tb.start()	666	
669 sum_change5 = 0 670 sum_change6 = 0 671 sum_change7 = 0 672 sum_change8 = 0 673 sum_change9 = 0 674 sum_change10 = 0 675 sum_change11 = 0 676 sum_change12 = 0 677 sum_change13 = 0 678 the true: 680 tb = text_rx() 681 tb.start() 682 data1 = open('/home/o/test_file2', 'r+')	667	
670 sum_change6 = 0 671 sum_change7 = 0 672 sum_change8 = 0 673 sum_change9 = 0 674 sum_change10 = 0 675 sum_change11 = 0 676 sum_change13 = 0 677 sum_change13 = 0 678 th = text_rx() 680 th = text_rx() 61 th = text_file2', 'r+')	668	
671 sum_change7 = 0 672 sum_change8 = 0 673 sum_change9 = 0 674 sum_change10 = 0 675 sum_change11 = 0 676 sum_change12 = 0 677 sum_change13 = 0 678 sum_change13 = 0 679 while True: 680 tb = text_rx() tb = text_rx() tb.start() 681 tb.start() 682 data1 = open('/home/o/test_file2', 'r+')	669	
672 sum_change8 = 0 673 sum_change9 = 0 674 sum_change10 = 0 675 sum_change11 = 0 676 sum_change12 = 0 677 sum_change13 = 0 678 sum_change13 = 0 679 while True: 680 tb = text_rx() 681 tb.start() 682 data1 = open('/home/o/test_file2', 'r+')		
<pre>673 sum_change9 = 0 674 sum_change10 = 0 675 sum_change11 = 0 676 sum_change12 = 0 677 sum_change13 = 0 678 679 while True: 680 tb = text_rx() 681 tb.start() 682 data1 = open('/home/a/test_file2', 'r+')</pre>	672	
674 sum_change10 = 0 675 sum_change11 = 0 676 sum_change12 = 0 677 sum_change13 = 0 678 sum_change13 = 0 679 while True: 680 tb = text_rx() 61 tb.start() 622 data1 = open('/home/a/test_file2', 'r+')	673	
676 sum_change12 = 0 677 sum_change13 = 0 678 this true: 680 tb = text_rx() 681 tb.start() 682 data1 = open('/home/o/test_file2', 'r+')	674	<pre>sum_change10 = 0</pre>
<pre>677 sum_change13 = 0 678 679 while True: 680</pre>	675	
678	676	
679 while True: 680 tb = text_rx() 681 tb.start() 682 datal = open('/home/o/test_file2', 'r+')		<pre>sum_changel3 = 0</pre>
680 tb = text_rx() 681 tb.start() 682 data1 = open('/home/c/test_file2', 'r+')		while True:
681 tb.start() 682 data1 = open('/home/c/test_file2', 'r+')	680	
<pre>data1 = open('/home/c/test_file2', 'r+')</pre>	681	
603 final = open('/home/c/final', 'a')	682	<pre>data1 = open('/home/c/test_file2', 'r+')</pre>
	683	<pre>final = open('/home/c/final', 'a')</pre>

684	while True:
685	diff = 0
686	data1.flush()
687	data1.readline()
688	<pre>file_content2 = data1.readline()</pre>
689	<pre>if (file_content2 == "eof\n"):</pre>
690 691	print(file_content2) break
692	Dreak elif (file_content2_!= ""):
693	$\frac{1}{1} \frac{1}{1} \frac{1}$
694	final.write(file content2)
695	<pre>split = file_content2.split(' ')</pre>
696	<pre>if (split[1].startswith('10') and (split[1].startswith('0', 2) == False)):</pre>
697	a_num += 1
698	actual_a_list.append(float(split[0]))
699 700	if $(\underline{a}, \underline{nun} > 1)$:
701	<pre>diff = abs(predict_a_list[-1]-actual_a_list[-1]) print(diff)</pre>
702	<pre>if (diff > 2*int(weight1)):</pre>
703	weight1 = int(weight1)+1
704	<pre>elif (int(weight1) > 1) and (diff < int(weight1)):</pre>
705	weight1 = int(weight1)-1
706	print(weight)
707 708	<pre>if (a_num > 1):</pre>
708	changev - actual_a_list[-1]-actual_a_list[-2] sum_change0 += change0
710	predict_a = actual_a list[-1] + sum_change0/a_num
711	else:
712	<pre>predict_a = actual_a_list[-1]</pre>
713	<pre>print("Prediction for Next A: "+str(predict_a))</pre>
714 715	predict a list.append(predict a)
715	<pre>elif (split[1].startswith('20')): b num += 1</pre>
717	actual b list.append(float(split[0]))
718	if $(b num \ge 1)$:
719	<pre>diff = abs(predict_b_list[-1]-actual_b_list[-1])</pre>
720	print(diff)
721 722	<pre>if (diff > 2*int(weight2)):</pre>
723	<pre>weight2 = int(weight2)+1 elif (int(weight2) > 1) and (diff < int(weight2)):</pre>
724	weight2 = int(weight2)-1
725	print(weight2)
726	if (b_num > 1):
727	<pre>change1 = actual_b_list[-1]-actual_b_list[-2]</pre>
728 729	<pre>sum_change1 += change1</pre>
729	<pre>predict_b = actual_b_list[-1] + sum_change1/b_num else:</pre>
731	predict_b = actual b_list[-1]
732	<pre>print("Prediction for Next B: "+str(predict_b))</pre>
733	predict_b_list.append(predict_b)
734	<pre>elif (split[1].startswith('30')):</pre>
735	c num += 1
736 737	<pre>actual_c_list.append(float(split[0])) if (c_num > 1):</pre>
738	<pre>diff = abs(predict_c_list[-1]-actual_c_list[-1])</pre>
739	print(diff)
740	<pre>if (diff > 2*int(weight3)):</pre>
741	weight3 = int(weight3)+1
742	<pre>elif (int(weight3) > 1) and (diff < int(weight3)):</pre>
743 744	<pre>weight3 = int(weight3)-1 print(weight3)</pre>
744	<pre>print(weights) if (c num > 1):</pre>
746	change2 = actual_c_list[-1]-actual_c_list[-2]
747	sum_change2 += change2
748	<pre>predict_c = actual_c_list[-1] + sum_change2/c_num</pre>
749	else:
750	predict_c = actual_c list[-1]
751 752	<pre>print("Prediction for Next C: "+str(predict_c)) predict_c_list.append(predict_c)</pre>
752	<pre>predict_c_list.append(predict_c) elif(appli(1).startswith('40')):</pre>
754	$d_{\text{num}} \neq 1$
755	actual_d_list.append(float(split[0]))
756	if (d_num > 1):
757	diff = abs(predict d list[-1]-actual d list[-1])

758	print (diff)
759	<pre>if (diff > 2*int(weight4)):</pre>
760	<pre>weight4 = int(weight4)+1</pre>
761	<pre>elif (int(weight4) > 1) and (diff < int(weight4)):</pre>
762	<pre>weight4 = int(weight4)-1</pre>
763	print(weight4)
764	if (<u>d_num > 1</u>):
765	<pre>change3 = actual_d_list[-1]-actual_d_list[-2]</pre>
766	<pre>sum_change3 += change3</pre>
767	<pre>predict_d = actual_d_list[-1] + sum_change3/d_num</pre>
768	else:
769	<pre>predict_d = actual_d_list[-1]</pre>
770	<pre>print("Prediction for Next D: "+str(predict_d))</pre>
771	predict_d_list.append(predict_d)
772	<pre>elif (split[1].startswith('50')):</pre>
773	e_num += 1
774	<pre>actual_e_list.append(float(split[0]))</pre>
775	if (e_num > 1):
776	<pre>diff = abs(predict_e_list[-1]-actual_e_list[-1])</pre>
777	print(diff)
778	<pre>if (diff > 2*int(weight5)):</pre>
779	<pre>weight5 = int(weight5)+1</pre>
780	<pre>elif (int(weight5) > 1) and (diff < int(weight5)):</pre>
781	weight5 = int(weight5)-1
782	print(weight5)
783	if $(e_n nm > 1)$:
784	<pre>change4 = actual_e_list[-1]-actual_e_list[-2]</pre>
785	sum_change4 += change4
786	<pre>predict_e = actual_e_list[-1] + sum_change4/e_num</pre>
787	else:
788	predict $e = actual e list[-1]$
789	<pre>print("Prediction for Next E: "+str(predict_e))</pre>
790	predict_e_list.append(predict_e)
791	<pre>elif (split[1].startswith('60')):</pre>
792	f num += 1
793	
794	if (f num > 1):
795	diff = abs(predict_f_list[-1]-actual_f_list[-1])
796	print(diff)
797	if (diff > 2*int(weight6)):
798	<pre>weight6 = int(weight6)+1</pre>
799	<pre>elif (int(weight6) > 1) and (diff < int(weight6)):</pre>
800	<pre>weight6 = int(weight6)-1</pre>
801	print(weight6)
802	if (f_num > 1):
803	<pre>change5 = actual_f_list[-1]-actual_f_list[-2]</pre>
804	<pre>sum_change5 += change5</pre>
805	<pre>predict_f = actual_f_list[-1] + sum_change5/f_num</pre>
806	else:
807	<pre>predict_f = actual_f_list[-1]</pre>
808	<pre>print("Prediction for Next F: "+str(predict_f))</pre>
809	predict_f_list.append(predict_f)
810	<pre>elif (split[1].startswith('70')):</pre>
811	g_num += 1
811 812	<pre>g_num += 1 actual_g_list.append(float(split[0]))</pre>
812	<pre>actual_g_list.append(float(split[0]))</pre>
812 813	<pre>actual_g_list.append(float(split[0])) if (g_num > 1): diff = abs(predict_g_list[-1]-actual_g_list[-1]) print(diff)</pre>
812 813 814	<pre>actual_g_list.append(float(split[0])) if (g_num > 1): diff = abs(predict_g_list[-1]-actual_g_list[-1])</pre>
812 813 814 815	<pre>actual_g_list.append(float(split[0])) if (g_num.>1): diff = abs(predict_g_list[-1]-actual_g_list[-1]) print(diff) if (diff > 2*int(weight7)): weight7 = int(weight7)+1</pre>
812 813 814 815 816	<pre>actual_g_list.append(float(split[0])) if (g_num > 1): diff = abs(predict_g_list[-1]-actual_g_list[-1]) print(diff) if (diff > 2*int(weight7)):</pre>
812 813 814 815 816 817	<pre>actual_g_list.append(float(split[0])) if (g_num.>1): diff = abs(predict_g_list[-1]-actual_g_list[-1]) print(diff) if (diff > 2*int(weight7)): weight7 = int(weight7)+1</pre>
812 813 814 815 816 817 818	<pre>actual_g_list.append(float(split[0])) if (g_num > 1): diff = abs(predict_g_list[-1]-actual_g_list[-1]) print(diff) if (diff > 2*int(weight7)): weight7 = int(weight7)+1 elif (int(weight7) > 1) and (diff < int(weight7)):</pre>
812 813 814 815 816 817 818 819	<pre>actual_g_list.append(float(split[0])) if (g_num > 1): diff = abs(predict_g_list[-1]-actual_g_list[-1]) print(diff) if (diff > 2*int(weight7)): weight7 = int(weight7))+1 elif (int(weight7) - 1) and (diff < int(weight7)): weight7 = int(weight7)-1</pre>
812 813 814 815 816 817 818 819 820 821 822	<pre>atual_g_list.append(float(split[0])) if (g_num > 1): diff = abs(predict_g_list[-1]-actual_g_list[-1]) print(diff) if (diff > 2*int(weight7)): weight7 = int(weight7)+1 elif (int(weight7) > 1) and (diff < int(weight7)): weight7 = int(weight7)-1 print(weight7) if (g_num > 1): change6 = actual_g_list[-1]-actual_g_list[-2] </pre>
812 813 814 815 816 817 818 819 820 821	<pre>actual_g_list.append(float(split[0])) if (g_num > 1): diff = abs(predict_g_list[-1]-actual_g_list[-1]) print(diff) if (diff > 2*int(weight7)): weight7 = int(weight7)+1 elif (int(weight7) > 1) and (diff < int(weight7)): weight7 = int(weight7)-1 print(weight7) if (g_num > 1):</pre>
812 813 814 815 816 817 818 819 820 821 822	<pre>atual_g_list.append(float(split[0])) if (g_num > 1): diff = abs(predict_g_list[-1]-actual_g_list[-1]) print(diff) if (diff > 2*int(weight7)): weight7 = int(weight7)+1 elif (int(weight7) > 1) and (diff < int(weight7)): weight7 = int(weight7)-1 print(weight7) if (g_num > 1): change6 = actual_g_list[-1]-actual_g_list[-2] </pre>
812 813 814 815 816 817 818 819 820 821 822 822 823	<pre>atual_g_list.append(float(split[0])) if (g_num > 1): diff = abs(predict_g_list[-1]-actual_g_list[-1]) print(diff) if (diff > 2*int(weight7)): weight7 = int(weight7)+1 elif (int(weight7) > 1) and (diff < int(weight7)): weight7 = int(weight7)-1 print(weight7) if (g_num > 1): change6 = actual_g_list[-1]-actual_g_list[-2] sum_change6 += change6</pre>
812 813 814 815 816 817 818 819 820 821 822 821 822 823 824	<pre>actual_g_list.append(float(split[0])) if (g_num_>l): diff = abs(predict_g_list[-1]-actual_g_list[-1]) print(diff) if (diff > 2*int(weight7)): weight7 = int(weight7)+1 elif (int(weight7) > 1) and (diff < int(weight7)): weight7 = int(weight7)-1 print(weight7) if (g_num_>l): change6 + actual_g_list[-1]-actual_g_list[-2] sum_change6 + actual_g_list[-1] + sum_change6/g_num </pre>
812 813 814 815 816 817 818 819 820 821 822 821 822 823 824 825	<pre>atual_g_list.append(float(split[0])) if (g_num > 1): diff = abs(predict_g_list[-1]-actual_g_list[-1]) print(diff) if (diff > 2*int(weight7)): weight7 = int(weight7)+1 elif (int(weight7) > 1) and (diff < int(weight7)): weight7 = int(weight7)-1 print(weight7) if (g_num > 1): change6 = actual_g_list[-1]-actual_g_list[-2] sum_change6 += change6 predict_g = actual_g_list[-1] + sum_change6/g_num else: </pre>
812 813 814 815 816 817 818 820 821 822 823 824 825 826	<pre>atual_g_list.append(float(split[0])) if (g_num > 1): diff = abs(predict_g_list[-1]-actual_g_list[-1]) print(diff) if (diff > 2*int(weight7)): weight7 = int(weight7)+1 elif (int(weight7) - 1) and (diff < int(weight7)): weight7 = int(weight7)-1 print(weight7) if (g_num > 1): change6 = actual_g_list[-1]-actual_g_list[-2] sum_change6 += change6 predict_g = actual_g_list[-1] + sum_change6/g_num else: predict_g = actual_g_list[-1]</pre>
812 813 814 815 816 817 818 820 821 822 823 824 825 826 827	<pre>actual_g_list.append(float(split[0])) if (g_num > 1): diff = abs(predict_g_list[-1]-actual_g_list[-1]) print(diff) if (diff > 2*int(weight7)): weight7 = int(weight7)+1 elif (int(weight7) > 1) and (diff < int(weight7)): weight7 = int(weight7)-1 print(weight7) if (g_num > 1): change6 = actual_g_list[-1]-actual_g_list[-2] sum_change6 += change6 predict_g = actual_g_list[-1] + sum_change6/g_num else: predict_g = actual_g_list[-1] print("Prediction for Next 6: "+str(predict_g))</pre>
812 813 814 815 816 817 818 820 821 822 823 824 825 826 827 828	<pre>atual_g_list.append(float(split[0])) if (g_num > 1): diff = abs(predict_g_list[-1]-actual_g_list[-1]) print(diff) if (diff > 2*int(weight7)): weight7 = int(weight7)+1 elif (int(weight7) > 1) and (diff < int(weight7)): weight7 = int(weight7)-1 print(weight7) if (g_num > 1): change6 = actual_g_list[-1]-actual_g_list[-2] sum_change6 += change6 predict_g = actual_g_list[-1] + sum_change6/g_num else: predict_g = actual_g_list[-1] print("Prediction for Next 6: "+str(predict_g)) predict_g_list.append(predict_g) elif (ieplit[1].startswith('80')): h_num *= 1 </pre>
812 813 814 815 816 817 818 819 820 821 822 823 824 825 826 827 828 829	<pre>atual_g_list.append(float(split[0])) if (g_num > 1): diff = abs(predict_g_list[-1]-actual_g_list[-1]) print(diff) if (diff > 2*int(weight7)): weight7 = int(weight7)+1 elif (int(weight7) + 1) and (diff < int(weight7)): weight7 = int(weight7)-1 print(weight7) if (g_num > 1): change6 = actual_g_list[-1]-actual_g_list[-2] sum_change6 += change6 predict_g = actual_g_list[-1] + sum_change6/g_num else: predict_g = actual_g_list[-1] print("Prediction for Next 6: "tstr(predict_g)) predict_glist.append(predict_g) elif (aplit[1].startswith('80')): </pre>

832	if $(h num > 1)$:
833	<pre>diff = abs(predict_h_list[-1]-actual_h_list[-1])</pre>
834	print(diff)
835 836	<pre>if (diff > 2*int(weight8)):</pre>
837	<pre>weight8 = int(weight8)+1</pre>
838	<pre>elif (int(weight8) > 1) and (diff < int(weight8)):</pre>
839	<pre>weight8 = int(weight8)-1 print(weight8)</pre>
840	if (h num > 1):
841	change7 = actual h_list[-1]-actual h_list[-2]
842	sum_change7 += change7
843	predict h = actual_h list[-1] + sum_change7/h_num
844	else:
845	predict h = actual_h list[-1]
846	print("Prediction for Next H: "+str(predict_h))
847	predict_h_list.append(predict_h)
848	<pre>elif (split[1].startswith('90')):</pre>
849	i num += 1
850	<pre>actual_i list.append(float(split[0]))</pre>
851	if (i_num > 1):
852	diff = abs(predict_i_list[-1]-actual_i_list[-1])
853	print(diff)
854	if (diff > 2*int(weight9)):
855	<pre>weight9 = int(weight9)+1</pre>
856	<pre>elif (int(weight9) > 1) and (diff < int(weight9)):</pre>
857	weight9 = int(weight9)-1
858	print(weight9)
859	if (<u>i_num > 1</u>):
860	<pre>change8 = actual_i_list[-1]-actual_i_list[-2]</pre>
861	<pre>sum_change8 += change8</pre>
862	<pre>predict_i = actual_i_list[-1] + sum_change8/i_num</pre>
863	else:
864	<pre>predict_i = actual_i_list[-1]</pre>
865	<pre>print("Prediction for Next I: "+str(predict_i))</pre>
866	predict_ilist.append(predict_i)
867	<pre>elif (split[1].startswith('100')):</pre>
868	<pre>j num += 1 actual j list.append(float(split[0]))</pre>
870	if $(j_{-num} > 1)$:
871	diff = abs(predict_j_list[-1]-actual_j_list[-1])
872	print(diff)
873	if (diff > 2*int(weight10)):
874	<pre>weight10 = int(weight10)+1</pre>
875	<pre>elif (int(weight10) > 1) and (diff < int(weight10)):</pre>
876	<pre>weight10 = int(weight10)-1</pre>
877	print (weight10)
878	if (j_num > 1):
879	<pre>change9 = actual_j_list[-1]-actual_j_list[-2]</pre>
880	<pre>sum_change9 += change9</pre>
881	<pre>predict_j = actual_j_list[-1] + sum_change9/j_num</pre>
882	else:
883	<pre>predict_j = actual_j_list[-1]</pre>
884	<pre>print("Prediction for Next J: "+str(predict_j))</pre>
885	predict_jlist.append(predict_j)
886	<pre>elif (split[1].startswith('110')):</pre>
887	$k \mod 1$
888	<pre>actual_k_list.append(float(split[0])) </pre>
889	if $(\underline{k} = \max)$:
890	<pre>diff = abs(predict_k_list[-1]-actual_k_list[-1]) print(diff)</pre>
891 892	<pre>print(diff) if (diff > 2*int(weight11)):</pre>
892	<pre>if (diff > 2*int(weight11)): weight11 = int(weight11)+1</pre>
894	<pre>weight1 = in(weight1) > 1) and (diff < int(weight1)):</pre>
895	weight1 = int(weight1) - 1
895	weightl = int(weightl) = i print(weightl)
897	if (k num > 1):
898	changel0 = actual k list[-1]-actual k list[-2]
899	sum changel0 += changel0
900	predict k = actual k list[-1] + sum_change10/k_num
901	else:
902	predict_k = actual_k_list[-1]
903	print("Prediction for Next K: "+str(predict k))
904	predict_k_list.append(predict_k)
905	<pre>elif (split[1].startswith('120'));</pre>

906 907	1 num + = 1
907	<pre>actual_llist.append(float(split[0])) <pre>if(""""""""""""""""""""""""""""""""""""</pre></pre>
908	<pre>if (l_num > 1): diff = abs(predict_l_list[-1]-actual_l_list[-1])</pre>
910	<pre>unit = abs[ptenteist[-1]=actualist[-1]) print(diff)</pre>
911	if (diff > 2*int(weight12)):
912	<pre>weight12 = int(weight12)+1</pre>
913	<pre>elif (int(weight12) > 1) and (diff < int(weight12)):</pre>
914	<pre>weight12 = int(weight12)-1</pre>
915	print(weight12)
916	if (1 num > 1):
917	<pre>change11 = actual_l_list[-1]-actual_l_list[-2]</pre>
918	sum_changel1 += changel1
919	<pre>predict_l = actual_l_list[-1] + sum_changel1/l_num</pre>
920	else:
921	<pre>predict_1 = actual_list[-1]</pre>
922	<pre>print("Prediction for Next L: "+str(predict_1))</pre>
923	<pre>predict_l_list.append(predict_l)</pre>
924	<pre>elif (split[1].startswith('130')):</pre>
925	m_num += 1
926	actual_m_list.append(float(split[0]))
927	if $(\underline{m}, \underline{num} > 1)$:
928	<pre>diff = abs(predict_m_list[-1]-actual_m_list[-1])</pre>
929	print(diff)
930	<pre>if (diff > 2*int(weight13)):</pre>
931	weight13 = int(weight13)+1
932 933	<pre>elif (int (weight13) > 1) and (diff < int (weight13)):</pre>
933 934	<pre>weight13 = int(weight13)-1 print(weight13)</pre>
935	if (m num > 1):
936	changel2 = actual m list[-1]-actual m list[-2]
937	sum_changel2 += changel2
938	predict_m = actual m_list[-1] + sum_change12/m_num
939	else:
940	predict_m = actual m_list[-1]
941	print("Prediction for Next M: "+str(predict m))
942	predict_m_list.append(predict_m)
943	<pre>elif (split[1].startswith('140'));</pre>
944	n_num += 1
945	<pre>actual_n_list.append(float(split[0]))</pre>
946	if $(n_n num \ge 1)$:
947	<pre>diff = abs(predict_n_list[-1]-actual_n_list[-1])</pre>
948	print(diff)
949	<pre>if (diff > 2*int(weight14)):</pre>
950 951	<pre>weight14 = int(weight14)+1 elif (int(weight14) > 1) and (diff < int(weight14)):</pre>
951	<pre>elif (int(weighti) / i) and (dif < int(weightig)): weightig = int(weightig)-1</pre>
953	wright = - in (wright = - i print(wright =)
954	if (n num > 1):
955	changel3 = actual n_list[-1]-actual_n_list[-2]
956	sum changel3 += changel3
957	predict n = actual n list[-1] + sum_change13/n num
958	else:
959	<pre>predict_n = actual_n_list[-1]</pre>
960	<pre>print("Prediction for Next N: "+str(predict_n))</pre>
961	predict_n_list.append(predict_n)
962	<pre>print("Current Value: "+file_content2)</pre>
963	time.sleep(1)
964	sb = ack()
965	sb.start()
966	print_"SENDING ACK"
967	time.sleep(1)
968	sb.stop()
969	if $(packet_tot > 28)$ and $(packet_tot \frac{4}{5} 20 = 0):$
970	<pre>weight_file = open('home/o/ohat3', 'w') wiibt_file = open('home/o/ohat3', 'w')</pre>
971	<pre>weight_file.write(str(weight) + ',') write(str(weight) + ',')</pre>
972	<pre>weight_file.write(str(weight2) + ',') writefile.write(str(weight2) + ', ')</pre>
973 974	<pre>weight_file.write(str(weight3) + ',') weight file.write(str(weight4) + ',')</pre>
974 975	<pre>weight_file.write(str(weight4) + ',') weight_file.write(str(weight5) + ',')</pre>
975	<pre>weight_file.write(str(weights) + ',') weight_file.write(str(weight6) + ',')</pre>
	<pre>weight_lite.write(str(weight0) + ',') weight file.write(str(weight0) + ',')</pre>
977	
977 978	weight file.write(str(weight8) + ',')

980	<pre>weight_file.write(str(weight10) + ',')</pre>
981	<pre>weight_file.write(str(weight11) + ', ')</pre>
982	<pre>weight_file.write(str(weight12) + ',')</pre>
983	<pre>weight_file.write(str(weight13) + ',')</pre>
984	weight file.write(str(weight14) + ',\n')
985	weight_file.close() () () () () ()
986	
	up = chat()
987	up.start()
988	time.sleep(1)
989	up.stop()
990	break
991	time.sleep(1)
992	
993	tb.stop()
994	
995	<pre>if (file_content2 == "eof\n"):</pre>
996	data1.close()
997	final.close()
998	break
999	
1000	data1.close()
1001	final.close()
1002	
1003	tb.stop()
1004	
1005	sum_sq_a = 0
1006	<pre>for z in range(0, len(predict_a_list)-1):</pre>
1007	<pre>square = (actual_a_list[z+1] - predict_a_list[z])**2</pre>
1008	sum sq a += square
1009	msgrterr_a = math.sgrt(sum_sq_a)
1010	print("Root Mean Squared Error for Sensor 1: "+str(msqrterr a))
1011	
1012	0 = d_pe_mue
1013	for z in range(0, len(predict_b_list)-1):
1014	<pre>square = (actual_b_list[z+1] - predict_b_list[z])**2</pre>
1015	sum sq b += square
1016	magriter b = math.sqrt (sum sq b)
1017	print("Root Mean Squared Error for Sensor 2: "+str(msqrterr_b))
1018	
1019	sum_sq_c = 0
1020	<pre>for z in range(0, len(predict_c_list)-1):</pre>
1021	square = (actual_c_list[z+1] - predict_c_list[z])**2
1022	sum_sq_c += square
1023	msqrter_c = math.sqrt(sum_sq_c)
1024	print("Root Mean Squared Brror for Sensor 3: "+str(msqrterr_c))
1025	
1026	sum_sq_d = 0
1027	for z in range(0, len(predict d list)-1):
1028	<pre>square = (actual_d_list[z+1] - predict_d_list[z])**2</pre>
1029	sum_sq_d += square
1030	msqrter_d = math.sqrt(sum_sq_d)
1031	print("Root Mean Squared Error for Sensor 4: "+str(msqrterr d))
1032	
1033	$sum_{sq} = 0$
1034	<pre>cm</pre> for z in range(0, len(predict_e_list)-1):
1035	square = (actual_elist[z1] - predict_e list[z])**2
1036	sum sq e += square
1037	msqrter: e = math.sqrt(sum_sq_e)
1038	print("Root Mean Squared Error for Sensor 5: "+str(msqrterr e))
1039	
1040	$sum_sq_f = 0$
1041	<pre>for z in range(0, len(predict f_list)-1):</pre>
1042	<pre>square (actual_flist[z+1] - predict_flist[z])**2</pre>
1043	sums = { (scourscours) { } } { } { } { } { } { } { } { } { }
1043	sum_sqlt square magrtesr f = math.sqrt(sum_sqlf)
1044	magneedr mach.sqf (sum_sq_r) print("Root Mean Squared Error for Sensor 6: "+str(msqrterr_f))
1045	1.1-1-1-1-1-1-1-1-1-1-1-1-1-1-1-1-1-1-1
1046	$\pi_{ij} = \sigma_{ij} = 0$
1047	<pre>sum_sq_g = 0 for z in range(0, len(predict_g_list)-1):</pre>
1048	
1049	<pre>square = (actual_g_list[z+1] - predict_g_list[z])**2 sym srg d t= sumsra</pre>
1050	sum sq g + square
	magrteer_g = math.sqrt(sum_sq_g)
1052 1053	print("Root Mean Squared Error for Sensor 7: "+str(msqrterr_g))
2000	

1054	
1054	<pre>sum_sq_h = 0 for z in range(0, len(predict h_list)-1):</pre>
1055	square = (actual h list[z+1] - predict h list[z])**2
1056	square - (acuaa_n_ist(z*i) - prencc_n_ist(z))2
1058	magtretr h = math.sqrt(sum_sq_h)
1059	print ("Root Mean Squared Error for Sensor 8: "+str(msqrterr_h))
1060	hence inter administration of a content of a content of the first of t
1061	pum sq i = 0
1062	for z in range(0, len(predict i_list)-1):
1063	<pre>square = (actual i list[z1] - predict i list[z])**2</pre>
1064	sum sq i += square
1065	msgrterr i = math.sqrt(sum_sq_i)
1066	print("Root Mean Squared Error for Sensor 9: "+str(msqrterr_i))
1067	
1068	$sum_{sq} = 0$
1069 🤇	for z in range(0, len(predict_j_list)-1):
1070	<pre>square = (actual_j_list[z+1] - predict_j_list[z])**2</pre>
1071	sum_sq_j += square
1072	<pre>msgrterr_j = math.sqrt(sum_sq_j)</pre>
1073	<pre>print("Root Mean Squared Error for Sensor 10: "+str(msqrterr_j))</pre>
1074	
1075	sum_sq_k = 0
1076	for z in range(0, len(predict_k_list)-1):
1077	<pre>square = (actual_k_list[z+1] - predict_k_list[z])**2</pre>
1078	sum_sq_k += square
1079	<pre>msgrterr_k = math.sqrt(sum_sq_k)</pre>
1080	<pre>print("Root Mean Squared Error for Sensor 11: "+str(msqrterr_k))</pre>
1081	
1082	sum_sq_l = 0
1083	<pre>o for z in range(0, len(predict_l_list)-1):</pre>
1084	<pre>square = (actual_l_list[z+1] - predict_l_list[z])**2</pre>
1085	e sum_sql += square
1086	mggrterr 1 = math.sqrt(sum sq 1)
1087	print("Root Mean Squared Error for Sensor 12: "+str(msqrterr_1))
1089	= m_pe_m = 0
1000	for z in range(0, len(predict m list)-1):
1091	square = (actual m list[z+1] - predict m_list[z])**2
1092	sum_sq_m += square
1093	msgrterr_m = math.sqrt(sum_sq_m)
1094	<pre>print("Root Mean Squared Error for Sensor 13: "+str(msqrterr_m))</pre>
1095	
1096	sum_sq_n = 0
1097	<pre>for z in range(0, len(predict_n_list)-1):</pre>
1098	<pre>square = (actual_n_list[z+1] - predict_n_list[z])**2</pre>
1099 (sum_sq_n += square
1100	<pre>msqrterr_n = math.sqrt(sum_sq_n)</pre>
1101	<pre>print("Root Mean Squared Error for Sensor 14: "+str(msqtterr_n))</pre>
1102	
1103	<pre>rmse_tot = msqrterr_a+msqrterr_b+msqrterr_c+msqrterr_d+msqrterr_e+msqrterr_f+msqrterr_g+msqrterr_h+msqrterr_i+msqrterr_j+msqrterr_k+msqrt</pre>
1104	print("\nTotal Root Mean Squared Error: "+str(rmse_tot))
1105	
1106 1107	exit()
1107	def mitting().
1108	<pre>def quitting(): tb.stop()</pre>
1109	cb.scop() cb.wait()
1110	<pre>qapp.connect(app, Qt.SIGNAL("aboutToQuit()"), quitting)</pre>
1112	dapp.exec_() dapp.exec_()
1112	
1113	
1115	ifname == 'main':
1116	main()

Figure A.3: Smart Weighted Round-Robin Algorithm for Base Station Radio

1		sr/bin/env python2
2	# -*	- coding: utf-8 -*-
3	####	
4	# GN	U Radio Python Flow Graph
5	# Ti	tle: Chat2
6	# Ge	nerated: Thu Feb 10 09:52:14 2022
7	0 ####	
8		
9 🕨	🤤 if 🔤	name == ' main ':
10		import ctypes
11		import sys
12		if sys.platform.startswith('linux'):
13		try:
14		<pre>x11 = ctypes.cdll.LoadLibrary('libX11.so')</pre>
15		x11.XInitThreads()
16		except:
17		print_"Warning: failed to XInitThreads()"
18		
19	impo	rt
39		
40		
41	clas	s chat2(gr.top_block, Qt.QWidget):
42		
43		definit(self):
44		<pre>gr.top_blockinit(self, "Chat2")</pre>
45		Qt.QWidgetinit(self)
46		self.setWindowTitle("Chat2")
47		qtgui.util.check_set_qss()
48		try:
49		<pre>self.setWindowIcon(Qt.QIcon.fromTheme('gnuradio-grc'))</pre>
50		except:
51		pass
52		<pre>self.top_scroll_layout = Qt.QVBoxLayout()</pre>
53		<pre>self.setLayout(self.top_scroll_layout)</pre>
54		<pre>self.top_scroll = Qt.QScrollArea()</pre>
55		self.top_scroll.setFrameStyle(Qt.QFrame.NoFrame)
56		self.top_scroll_layout.addWidget(self.top_scroll)
57		self.top_scroll.setWidgetResizable(True)
58		<pre>self.top_widget = Qt.QWidget()</pre>
59		self.top_scroll.setWidget(self.top_widget)
60		<pre>self.top_layout = Qt.QVBoxLayout(self.top_widget)</pre>
61		<pre>self.top_grid_layout = Qt.QGridLayout()</pre>
62		<pre>self.top_layout.addLayout(self.top_grid_layout)</pre>
63		
64		<pre>self.settings = Qt.QSettings("GNV Radio", "chat2")</pre>
65		<pre>self.restoreGeometry(self.settings.value("geometry").toByteArray())</pre>
66		
67		
68		
69		# Variables
70		
71		self.samp_rate = samp_rate = 400000
72		
73		
74		# Blocks
75		
76		<pre>self.uhd_usrp_source() = uhd.usrp_source(</pre>
77		",".join(("", "")),
78		uhd.stream_args(
79		cpu_format="fo32",
80		<pre>channels=range(1),</pre>
81		
82		
83 84		self.uhd_usrp_source_0.set_samp_rate(samp_rate)
		self.uhd_usrp_source_0.set_center_freq(100000000, 0)
85		<pre>self.uhd_usrp_source_0.set_gain(30, 0) self.uhd_usrp_source_0.set_antenna('TX/RX', 0)</pre>
86		
87 88		<pre>self.gtgui_freq_sink_x_0 = qtgui.freq_sink_c(</pre>
		1024, faire
89		firdes.HIN_BLACKMAN_hARRIS, #vintype
90		0, ffo
		samp_rate, #bw
91		
91 92 93		"", fname 1 fnumber of inputs

94)	
95	self.qtgui_freq_sink x_0.set_update_time(0.10)	
96	self.qtgui freq sink x 0.set y axis(-140, 10)	
97	self.qtgui_freq_sink_x_0.set_y_label('Relative Gain', 'dB')	
98	self.qtgui_freq_sink_x_0.set_trigger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "")	
99	self.qtgui_freq_sink_x_0.enable_autoscale(False)	
100	self.qtgui_freq_sink_x_0.enable_grid(False)	
101	self.qtgui_freq_sink_x_0.set_fft_average(1.0)	
102	self.qtgu_freq_sink_x_0.enable_axis_labels(True)	
103	self.qtgu_freq_sink_x_0.enable_control_panel (False)	
104		
105	if not True:	
106	self.qtgui freq sink x 0.disable legend()	
107	······································	
108	if "complex" == "float" or "complex" == "msg float":	
109	self.qtgui_freq_sink x_0.set plot_pos_half(not True)	
110		
111	labels = ['', '', '', '', '',	
112		
113	widths = [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1	
114	1, 1, 1, 1, 1	
115	colors = ["blue", "red", "green", "black", "cyan",	
116	"magenta", "yellow", "dark red", "dark green", "dark blue"]	
117	alphas = [1.0, 1.0, 1.0, 1.0, 1.0,	
118	1.0, 1.0, 1.0, 1.0, 1.0]	
119	<pre>for i in xrange(1):</pre>	
120	if len(labels[i]) == 0:	
121	<pre>self.qtgui_freq_sink_x_0.set_line_label(i, "Data {0}".format(i))</pre>	
122	else:	
123	<pre>self.qtgui_freq_sink_x_0.set_line_label(i, labels[i])</pre>	
124	self.qtgui_freq_sink x_0.set_line_width(i, widths[i])	
125	self.qtgui freq sink x 0.set line_color(i, colors[i])	
126	<pre>self.qtgui freq sink x 0.set line alpha(i, alphas[i])</pre>	
127		
128	self. qtgui freq sink x 0 win = sip.wrapinstance(self.qtgui freq sink x 0.pyqwidget(), Qt.QWidget)	
129	self.top grid layout.addWidget(self. qtgui freq sink x 0 win)	
130	self.low pass filter 0 = filter.fir filter ccf(1, firdes.low pass(
131	1, samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76))	
132	self.digital_gmsk_demod_1 = digital.gmsk_demod(
133	<pre>samples_per_symbol=2,</pre>	
134	gain_mu=0.175,	
135	mu=0.5,	
136	<pre>omega_relative_limit=0.005,</pre>	
137	<pre>freq_error=0.0,</pre>	
138	verbose=False,	
139	log=False,	
140)	
141	<pre>self.blocks_multiply_const_vxx_1 = blocks.multiply_const_vcc((1,))</pre>	
142	<pre>self.blocks_file_sink_1 = blocks.file_sink(gr.sizeof_char*1, '/home/b/chat_test1.txt', False)</pre>	
143	<pre>self.blocks_file_sink_1.set_unbuffered(False)</pre>	
144	<pre>self.blks2_packet_decoder_1 = grc_blks2.packet_demod_b(grc_blks2.packet_decoder(</pre>	
145	access_code='',	
146	threshold=-1,	
147	callback=lambda ok, payload: self.blks2_packet_decoder_1.recv_pkt(ok, payload),	
148),	
149	1	
150		
151		
152		
153		
154	# Connections	
155		
156	<pre>self.connect((self.blks2_packet_decoder_1, 0), (self.blocks_file_sink_1, 0))</pre>	
157	<pre>self.connect((self.blocks_multiply_const_vxx_1, 0), (self.low_pass_filter_0, 0))</pre>	
158	<pre>self.connect((self.digital_gmsk_demod_1, 0), (self.blks2_packet_decoder_1, 0))</pre>	
159	<pre>self.connect((self.low_pass_filter_0, 0), (self.digital_gmsk_demod_1, 0))</pre>	
160	<pre>self.connect((self.low_pass_filter_0, 0), (self.qtgui_freq_sink_x_0, 0))</pre>	
161	<pre>self.connect((self.uhd_usrp_source_0, 0), (self.blocks_multiply_const_vxx_1, 0))</pre>	
162		
163	<pre>def closeEvent(self, event):</pre>	
164	<pre>self.settings = Qt.QSettings("GNU Radio", "chat2") self settings = setVelue("seeretw", self seve?seretw())</pre>	
165	<pre>self.settings.setValue("geometry", self.saveGeometry()) state</pre>	
166 167	event.accept()	
2.01		

168	<pre>def get_samp_rate(self):</pre>	
169	return self.samp_rate	
170 171		
171	<pre>def set_samp_rate(self, samp_rate): self.samp_rate = samp_rate</pre>	
172	self.udd usrp source 0.set samp rate(self.samp rate)	
174	self.qtyui freq sink x 0.set frequency range(0, self.samp rate)	
175	aslflow pass_filter_0.set_taps(firdes.low pass(1, self.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76))	
176		
177		
178	class text_tx(gr.top_block):	
179		
180	<pre>definit(self):</pre>	
181	gr.top_blockinit(self, "Text Tx")	
182		
183		
184	# Variables	
185		
186 187	<pre>self.samp_rate = samp_rate = 400000 source = ''</pre>	
188	Source	
189		
190	# Blocks	
191		
192	<pre>self.uhd_usrp_sink_0 = uhd.usrp_sink(</pre>	
193	",".join(("", "")),	
194	uhd.stream_args(
195	cpu_format="fo32",	
196	channels=range(1),	
197).	
198		
199 200	<pre>self.uhd_usrp_sink_0.set_samp_rate(samp_rate) self.uhd_usrp_sink_0.set_time_now(uhd.time_spec(time.time()), uhd.ALL_MBOARDS)</pre>	
200	<pre>sel.und_usrp_sink_0.set_enter_freq(100000000, 0) self.und_usrp_sink_0.set_enter_freq(100000000, 0)</pre>	
202	self.uduurp sink 0.set_gain(40, 0)	
203	self.uhd.usrp sink 0.set antenna('TX/RX', 0)	
204	self.digital_gmsk_mod_0 = digital.gmsk_mod(
205	<pre>samples_per_symbol=2,</pre>	
206	bt=0.35,	
207	verbose=False,	
208	log=False,	
209) self.blocks_throttle_0 = blocks.throttle(gr.sizeof_char*1, samp_rate,True)	
211	<pre>self.blocks_multiply_const_vax_0 = blocks.multiply_const_vac((1,))</pre>	
212	self.blocks_head_0 = blocks.head(gr.sizeof_char*1, samp_rate/4)	
213		
214	<pre>send = open('/home/b/tx.txt', 'w')</pre>	
215	if switch == 'a':	
216	<pre>send.write(data0[a1])</pre>	
217	<pre>print(data0[a1])</pre>	
218	elif switch == 'b':	
219 220	send.write(data1[b1]) print(data1[b1])	
220	<pre>print(data[0]) d elif switch == 'o':</pre>	
222	send write (data2[c1])	
223	print(data2[c1])	
224	elif switch == 'd':	
225	send.write(data3[d1])	
226	print(data3[d1])	
227	elif switch == 'e':	
228	send.write(data4[e1])	
229	<pre>print(data4[e1]) elif switch == 'f':</pre>	
230 231	elif switch == 'f': send.write(data5[f1])	
231	sena.write(data5[1]) print(data5[1])	
233	elif switch == 'g':	
234	send.write(data6[g1])	
235	print(data6[g1])	
236	elif switch == 'h':	
237	send.write(data7[h1])	
238	print(data7[h1])	
239	elif switch == 'i':	
240	send.write(data8[i1])	
241	print(data8[i1])	

elif switch == 'j': 242 243 send.write(data9[j1]) 244 print(data9[j1]) 245 elif switch == 'k': send.write(data10[k1]) print(data10[k1]) elif switch == 'l': send.write(data11[11]) print(data11[11]) elif switch == 'm': send.write(data12[m1]) print(data12[m1]) elif switch == 'n': send.write(data13[n1]) 256 print(data13[n1]) elif switch == 'eof': send.write("eof\n") print "eof" send.close() self.blocks_file_source_0 = blocks.file_source(gr.sizeof_char*1, '/home/b/tx.txt', True) 263 self.blocks_file_source_0.set_begin_tag(pmt.PMT_NIL)
self.blks2_packet_encoder_0 = grc_blks2.packet_mod_b(grc_blks2.packet_encoder(264 samples_per_symbol=2, 266 267 bits_per_symbol=1, preamble=''. access_code='', pad_for_usrp=True,), payload_length=0,) ***** # Con ****** self.connect((self.blks2_packet_encoder_0, 0), (self.digital_gmsk_mod_0, 0)) self.connect((self.blocks_file_source_0, 0), (self.blocks_thotble_0, 0))
self.connect((self.blocks_head_0, 0), (self.bls2_packet_encoder_0, 0)) self.connect((self.blocks_multiply_const_vxx_0, 0), (self.uhd_usrp_sink_0, 0)) self.connect((self.blocks throttle 0, 0), (self.blocks head 0, 0)) self.connect((self.digital_gmsk_mod_0, 0), (self.blocks_multiply_const_vxx_0, 0)) def get_samp_rate(self): return self.samp_rate 290 def set_samp_rate(self, samp_rate): 291 self.samp_rate = samp_rate self.uhd_usrp_sink_0.set_samp_rate(self.samp_rate) self.blocks_throttle_0.set_sample_rate(self.samp_rate) 294 self.blocks_head_0.set_length(self.samp_rate/4) 295 class WRRScheduler(): 298 cw = 0i = -1 data_set = [] max_s = None gcd_s = None len_s = None counter = {} def __init__(self, s = None): self._init_dataset(s) def _init_dataset(self, s): self.data_set = s
self.max_s = max(s, key=lambda x: x[1])[1] self.gcd_s = reduce(fractions.ged, [weight for data, weight in s]) self.len_s = len(s)

316		def schedule(self):	
317		while True:	
318		<pre>self.i = (self.i + 1) % self.len_s if self.i == 0;</pre>	
319 320		<pre>self.u == 0: self.cw = self.cw - self.gcd s</pre>	
321		$set (w - set (w - set (y) d_s))$	
322		self.cw = self.max_s	
323		if self.cw == 0:	
324		return None	
325		<pre>if self.data_set[self.i][1] >= self.cw:</pre>	
326		<pre>selfinc_counter(self.data_set[self.i])</pre>	
327		return self.data set[self.i]	
328			
329		<pre>def _inc_counter(self, item):</pre>	
330		try:	
331		<pre>self.counter[item[0]] += 1</pre>	
332		except KeyError:	
333		<pre>self.counter[item[0]] = 1</pre>	
334			
335 336		<pre>def set data(self, s): self.reset()</pre>	
337		<pre>selfinit_dataset(s)</pre>	
338		sontnrt_uataset(s)	
339		<pre>def reset_counter(self):</pre>	
340		<pre>self.counter = {}</pre>	
341			
342		def reset (self):	
343		self.cw = 0	
344		self.i = -1	
345		<pre>self.data_set = []</pre>	
346		self.max_s = None	
347		self.gcd_s = None	
348		self.len_s = None	
349		self.reset_counter()	
350 351		<pre>def get_next(self, n = 1):</pre>	
352		if n > 1:	
353		<pre>return [self.schedule() for i in range(0,n)]</pre>	
354		<pre>return self.schedule()</pre>	
355			
356			
357	cla	<pre>iss ack2(gr.top_block, Qt.QWidget):</pre>	
358			
359 360		definit(self):	
361		ar ten bleck init (colf "Chat2")	
362		<pre>gr.top_blockinit(self, "Chat2") Ot OWidgetinit(self)</pre>	
363		Qt.QWidgetinit(self)	
		Qt.QWidgetinit(self) self.setWindowTitle("Chat2")	
364		Qt.QWidgetinit(self)	
364 365		<pre>Qt.QWidgetinit(self) self.setWindowTitle("Chat2") qtgui.util.check_set_qss()</pre>	
		<pre>Qt.OWidgetinit(self) self.setWindowTitle("Chat2") qtgui.util.check_set_qss() try:</pre>	
365 366 367		<pre>Qt.QWidgetinit(self) self.setWindowTitle("Chat2") qtgui.util.check_set_qss() try: self.setWindowIcon(Qt.QIcon.fromTheme('gnuradio-gre')) except: pass</pre>	
365 366 367 368		<pre>Qt.QWidgetinit(self) self.setWindowTitle("Chat2") qtgui.util.check_set_qss() try: self.setWindowIcon(Qt.QIcon.fromTheme('gnuradio-grc')) except: pass self.top_scroll_layout = Qt.QVBoxLayout()</pre>	
365 366 367 368 369		<pre>Qt.QWidgetinit(self) self.setWindowTitle("Chat2") qtgui.util.check_set_qss() try: self.setWindowIcon(Qt.QIcon.fromTheme('gnuradio-grc')) except: pas self.top_scroll_layout = Qt.QVBoxLayout() self.setLayout(self.top_scroll_layout)</pre>	
365 366 367 368 369 370		<pre>Qt.QWidgetinit(self) self.setWindowTitle("Chat2") qtgui.util.check_set_qss() try: self.setWindowTcon(Qt.QIcon.fromTheme('gnuradio-gre')) except: pass self.top_scroll_layout = Qt.QVBoxLayout() self.top_scroll_layout (self.top_scroll_layout) self.top_scroll = Qt.QScrollArea()</pre>	
365 366 367 368 369 370 371		<pre>Qt.QWidgetinit(self) self.setWindowTitle("Chat2") qtgui.util.check_set_qss() try: self.setWindowTcon(Qt.QIcon.fromTheme('gnuradio-grc')) except: pass self.top_scroll_layout = Qt.QVBoxLayout() self.top_scroll_layout = Qt.QVBoxLayout() self.setEayout(self.top_scroll_layout) self.top_scroll_setFrameStyle(Qt.QFrame.NoFrame)</pre>	
365 366 367 368 369 370 371 371		<pre>Qt.QWidgetinit(self) self.setWindowTitle("Chat2") qtgui.util.check_set_qss() try: self.setWindowIcon(Qt.QIcon.fromTheme('gnuradio-grc')) except: pass self.top_scroll_layout = Qt.QVBoxLayout() self.setLayout(self.top_scroll_layout) self.top_scroll_eqt.QScrollArea() self.top_scroll_setFrameStyle(Qt.QFrame.NoFrame) self.top_scroll_layout.addWidget(self.top_scroll)</pre>	
365 366 367 368 369 370 371 372 373		<pre>Qt.OWidgetinit(self) self.setWindowTitle("Chat2") qtgui.util.check_set_qss() try: self.setWindowIcon(Qt.QIcon.fromTheme('gnuradio-grc')) except: pass self.top_scroll_layout = Qt.QVBoxLayout() self.setLayout(self.top_scroll_layout) self.top_scroll = Qt.QScrollArea() self.top_scroll = Qt.QScrollArea() self.top_scroll_layout.ayout(self.top_scroll) self.top_scroll.setWidgetResizable(True)</pre>	
365 366 367 368 369 370 371 372		<pre>Qt.QWidgetinit(self) self.setWindowTitle("Chat2") qtgui.util.check_set_qss() try: self.setWindowTcon(Qt.QIcon.fromTheme('gnuradio-grc')) except: pass self.top_scroll_layout = Qt.QVBoxLayout() self.top_scroll_layout = Qt.QVBoxLayout() self.top_scroll = Qt.QScrollArea() self.top_scroll = Qt.QScrollArea() self.top_scroll_setFrameStyle(Qt.QFrame.NoFrame) self.top_scroll_setWidget(self.top_scroll) self.top_scroll_setWidget(self.top_scroll) self.top_scroll_setWidget(self.top_scroll) self.top_scroll_setWidget(self.top_scroll)</pre>	
365 366 367 368 369 370 371 372 373 373		<pre>Qt.OWidgetinit(self) self.setWindowTitle("Chat2") qtgui.util.check_set_qss() try: self.setWindowIcon(Qt.QIcon.fromTheme('gnuradio-grc')) except: pass self.top_scroll_layout = Qt.QVBoxLayout() self.setLayout(self.top_scroll_layout) self.top_scroll = Qt.QScrollArea() self.top_scroll = Qt.QScrollArea() self.top_scroll_layout.ayout(self.top_scroll) self.top_scroll.setWidgetResizable(True)</pre>	
365 366 367 368 369 370 371 372 373 374 375		<pre>Qt.QWidgetinit(self) self.setWindowTitle("Chat2") qtgui.util.check_set_qss() try: self.setWindowTcon(Qt.QIcon.fromTheme('gnuradio-gre')) except: pass self.top_scroll_layout = Qt.QVBoxLayout() self.top_scroll_layout = Qt.QVBoxLayout() self.setEayout(self.top_scroll_layout) self.top_scroll.setFrameStyle(Qt.OFrame.NoFrame) self.top_scroll.setFrameStyle(Qt.OFrame.NoFrame) self.top_scroll_layout.addWidget(self.top_scroll) self.top_widget = Qt.QWidget(Self.top_scroll) self.top_widget = Qt.QWidget(Self.top_widget)</pre>	
365 366 367 368 369 370 371 372 373 374 375 376		<pre>Qt.QWidgetinit(self) self.setWindowTitle("Chat2") qtgui.util.check_set_qss() try: self.setWindowIcon(Qt.QIcon.fromTheme('gnuradio_grc')) except: pass self.top_scroll_layout = Qt.QVBoxLayout() self.setLayout(self.top_scroll_layout) self.top_scroll = Qt.QScrollArea() self.top_scroll_setFrameStyle(Qt.QFrame.NoFrame) self.top_scroll_setWidgetResizable(True) self.top_scroll.setWidgetResizable(True) self.top_scroll.setWidget(self.top_widget) self.top_layout = Qt.QVBoxLayout(self.top_widget) self.top_layout = Qt.QVBoxLayout(self.top_widget)</pre>	
365 366 367 368 370 371 372 373 374 375 375 376 377 378 379		<pre>Qt.QWidgetinit(self) self.setWindowTitle("Chat2") qtgui.util.check_set_qss() try: self.setWindowTcon(Qt.QIcon.fromTheme('gnuradio-grc')) except: pass self.top_scroll_layout = Qt.QVBoxLayout() self.setLayout(self.top_scroll_layout) self.top_scroll.setFrameStyle(Qt.QFrame.NoFrame) self.top_scroll_setFrameStyle(Qt.QFrame.NoFrame) self.top_scroll_setWidgetResizable(True) self.top_scroll.setWidget(self.top_widget) self.top_layout = Qt.QVBoxLayout(self.top_widget) self.top_grid_layout = Qt.QVBoxLayout() self.top_grid_layout = Qt.QVBoxLayout() self.top_layout = Qt.QVBoxLayout(self.top_widget) self.top_layout = Qt.QVBoxLayout(self.top_widget) self.top_grid_layout = Qt.QVBoxLayout(self.top_widget) self.top_layout = Qt.QVBoxLayout(self.top_grid_layout)</pre>	
365 366 367 368 370 371 372 373 374 375 376 377 378 379 379 379		<pre>Qt.QWidgetinit(self) self.setWindowTitle("Chat2") qtgui.util.check_set_qss() try: self.setWindowTcon(Qt.QIcon.fromTheme('gnuradio-gre')) except: pass self.top_scroll_layout = Qt.QVBoxLayout() self.setLayout(self.top_scroll_layout) self.top_scroll = Qt.QScrollArea() self.top_scroll_setVingetResizable(True) self.top_scroll_setWidgetResizable(True) self.top_layout = Qt.QGridLayout() self.top_layout = Qt.QGridLayout() self.top_layout = Qt.QGridLayout() self.top_layout.addMidget(self.top_widget) self.top_layout = Qt.QGridLayout() self.top_layout.addLayout(self.top_widget) self.top_layout.addLayout(self.top_widget) self.setLayout.addLayout(self.top_grid_layout) self.setLayout.addLayout(self.top_grid_layout)</pre>	
365 366 367 368 370 371 372 373 374 375 376 377 378 379 380 381		<pre>Qt.QWidgetinit(self) self.setWindowTitle("Chat2") qtgui.util.check_set_qss() try: self.setWindowTcon(Qt.QIcon.fromTheme('gnuradio-grc')) except: pass self.top_scroll_layout = Qt.QVBoxLayout() self.setLayout(self.top_scroll_layout) self.top_scroll.setFrameStyle(Qt.QFrame.NoFrame) self.top_scroll_setFrameStyle(Qt.QFrame.NoFrame) self.top_scroll_setWidgetResizable(True) self.top_scroll.setWidget(self.top_widget) self.top_layout = Qt.QVBoxLayout(self.top_widget) self.top_grid_layout = Qt.QVBoxLayout() self.top_grid_layout = Qt.QVBoxLayout() self.top_layout = Qt.QVBoxLayout(self.top_widget) self.top_layout = Qt.QVBoxLayout(self.top_widget) self.top_grid_layout = Qt.QVBoxLayout(self.top_widget) self.top_layout = Qt.QVBoxLayout(self.top_grid_layout)</pre>	
365 366 367 369 370 371 372 373 374 375 375 376 377 378 379 380 381 382		<pre>Qt.QWidgetinit(self) self.setWindowTitle("Chat2") qtgui.util.check_set_qss() try: self.setWindowTcon(Qt.QIcon.fromTheme('gnuradio-gre')) except: pass self.top_scroll_layout = Qt.QVBoxLayout() self.setLayout(self.top_scroll_layout) self.top_scroll = Qt.QScrollArea() self.top_scroll_setVingetResizable(True) self.top_scroll_setWidgetResizable(True) self.top_layout = Qt.QGridLayout() self.top_layout = Qt.QGridLayout() self.top_layout = Qt.QGridLayout() self.top_layout.addMidget(self.top_widget) self.top_layout = Qt.QGridLayout() self.top_layout.addLayout(self.top_widget) self.top_layout.addLayout(self.top_widget) self.setLayout.addLayout(self.top_grid_layout) self.setLayout.addLayout(self.top_grid_layout)</pre>	
365 366 367 370 371 372 373 374 375 376 377 378 379 380 381 382 383		<pre>Qt.QWidgetinit(self) self.setWindowTitle("Chat2") qtgui.util.check_set_qss() try: self.setWindowTcon(Qt.QIcon.fromTheme('gnuradio-gre')) except: pas self.top_scroll_layout = Qt.QVBoxLayout() self.top_scroll_layout = Qt.QVBoxLayout() self.top_scroll_setFrameStyle(Qt.QFrame.NoFrame) self.top_scroll.setWindgetResizable(True) self.top_widget = Qt.QWidget() self.top_layout = Qt.QWidget() self.top_layout = Qt.QWidget() self.top_layout = Qt.QWidget() self.top_layout.addMayout(self.top_widget) self.top_layout.addLayout(self.top_widget) self.top_layout.addLayout(self.top_widget) self.top_layout.addLayout(self.top_widget) self.top_layout.addLayout(self.top_widget) self.top_layout.addLayout(self.top_widget) self.top_layout.addLayout(self.top_yrid_layout) self.settings = Qt.QSettings("GNV Radio", "chat2") self.restoreGeometry(self.settings.value("geometry").toByteArray())</pre>	
365 366 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 383		<pre>Qt.QWidgetinit(self) self.setWindowTitle("Chat2") qtgui.util.check_set_qss() try: self.setWindowTcon(Qt.QIcon.fromTheme('gnuradio-gre')) except: pass self.top_scroll_layout = Qt.QWBOxLayout() self.top_scroll = Qt.QWBOxLayout() self.top_scroll = Qt.QWBOxLayout() self.top_scroll = Qt.QScrollArea() self.top_scroll.setFrameStyle(Qt.QFrame.NoFrame) self.top_scroll.setWidgetResizable(True) self.top_scroll.setWidgetResizable(True) self.top_scroll.setWidget(self.top_widget) self.top_arcoll.setWidget(self.top_widget) self.top_drout = Qt.QWBOxLayout() self.top_drout = Qt.QWBOxLayout() self.top_arcoll.setWidget(self.top_widget) self.top_drout = Qt.QWBOxLayout() self.top_drout = Qt.QWBOxLayout() self.top_layout = Qt.QWBOxLayout() self.top_layout = Qt.QWBOxLayout() self.settings = Qt.QSettings("GNU Radio", "chat2") self.restoreGecmetry(self.settings.value("geometry").toByteArray()) </pre>	
365 366 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 384 385		<pre>Qt.QWidgetinit(self) self.setWindowTitle("Chat2") qtgui.util.check_set_qss() try: self.setWindowTcon(Qt.QIcon.fromTheme('gnuradio-gre')) except: pas self.top_scroll_layout = Qt.QVBoxLayout() self.top_scroll_layout = Qt.QVBoxLayout() self.top_scroll_setFrameStyle(Qt.QFrame.NoFrame) self.top_scroll.setWindgetResizable(True) self.top_widget = Qt.QWidget() self.top_layout = Qt.QWidget() self.top_layout = Qt.QWidget() self.top_layout = Qt.QWidget() self.top_layout.addMayout(self.top_widget) self.top_layout.addLayout(self.top_widget) self.top_layout.addLayout(self.top_widget) self.top_layout.addLayout(self.top_widget) self.top_layout.addLayout(self.top_widget) self.top_layout.addLayout(self.top_widget) self.top_layout.addLayout(self.top_yrid_layout) self.settings = Qt.QSettings("GNV Radio", "chat2") self.restoreGeometry(self.settings.value("geometry").toByteArray())</pre>	
365 366 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 383		<pre>Qt.QWidgetinit(self) self.setWindowTitle("Chat2") qtgui.util.check_set_qss() try: self.setWindowTon(Qt.QIcon.fromTheme('gnuradio-gro')) except: pass self.top_scroll_layout = Qt.QVBoxLayout() self.setTayout(self.top_scroll_layout) self.top_scroll.setFrameStyle(Qt.QFrame.NoFrame) self.top_scroll_setWidgetResizable(True) self.top_scroll.setWidgetResizable(True) self.top_ayout = Qt.QVBoxLayout() self.top_ayout = Qt.QVBoxLayout(self.top_widget) self.top_ayout = Qt.QVBoxLayout(self.top_widget) self.top_layout = Qt.QVBoxLayout(self.top_widget) self.top_layout.addLayout(self.top_grid_layout) self.settings = Qt.QSettings("GNU Radio", "chat2") self.restoreGeometry(self.settings.value("geometry").toByteArray()) ###################################</pre>	
365 366 369 370 371 372 373 374 375 376 377 377 377 378 377 378 379 381 381 382 383 382 383 384 385 385		<pre>Qt.QWidgetinit(self) self.setWindowTitle("Chat2") qtgui.util.check_set_qss() try: self.setWindowTon(Qt.QTcon.fromTheme('gnuradio-grc')) except: pass self.top_scroll_layout = Qt.QVBoxLayout() self.top_scroll_layout = Qt.QVBoxLayout() self.top_scroll_setFrameStyle(Qt.QFrame.NoFrame) self.top_scroll_ayout.addWidget(self.top_scroll) self.top_scroll_layout.addWidget(self.top_scroll) self.top_scroll_setWidget(self.top_widget) self.top_scroll.setWidget(self.top_widget) self.top_groll.setWidget(self.top_widget) self.top_groll.setWidget(self.top_grid_layout) self.top_grid_layout = Qt.QVBoxLayout() self.top_grid_layout = Qt.QVBoxLayout() self.top_layout = Qt.QVBoxLayout(self.top_widget) self.top_layout = Qt.QVBoxLayout(self.top_widget) self.top_layout = Qt.QVBoxLayout(self.top_widget) self.top_layout(self.top_grid_layout) self.settings = Qt.QSettings("GNV Radio", "chat2") self.restoreGeometry(self.settings.value("geometry").toByteArray()) ###################################</pre>	
365 366 369 370 371 372 373 374 375 375 376 377 378 379 380 381 381 382 383 384 385 384 385 386 386 387		<pre>Qt.QWidgetinit(self) self.setWindowTitle("Chat2") qtgui.util.check_set_qss() try: self.setWindowTon(Qt.QIcon.fromTheme('gnuradio-gro')) except: pass self.top_scroll_layout = Qt.QVBoxLayout() self.setTayout(self.top_scroll_layout) self.top_scroll.setFrameStyle(Qt.QFrame.NoFrame) self.top_scroll_setWidgetResizable(True) self.top_scroll.setWidgetResizable(True) self.top_ayout = Qt.QVBoxLayout() self.top_ayout = Qt.QVBoxLayout(self.top_widget) self.top_ayout = Qt.QVBoxLayout(self.top_widget) self.top_layout = Qt.QVBoxLayout(self.top_widget) self.top_layout.addLayout(self.top_grid_layout) self.settings = Qt.QSettings("GNU Radio", "chat2") self.restoreGeometry(self.settings.value("geometry").toByteArray()) ###################################</pre>	

```
390
                  # Blocks
                  self.uhd_usrp_source_0 = uhd.usrp_source(
    ",".join(("", "")),
                      uhd.stream_args(
                          cpu_format="fc32",
                          channels=range(1),
                      ),
                  self.uhd_usrp_source_0.set_samp_rate(samp_rate)
                  self.uhd_usrp_source_0.set_center_freq(1000000000, 0)
self.uhd_usrp_source_0.set_gain(30, 0)
402
                  self.uhd_usrp_source_0.set_antenna('TX/RX', 0)
                  self.gtgui_freq_sink_x_0 = qtgui.freq_sink_c(
                      1024, #size
                      firdes.WIN_BLACKMAN_hARRIS, #wintype
                      0, #fo
                      samp_rate, #bw
                      "", #name
                      1 #number of inputs
                  self.qtgui_freq_sink_x_0.set_update_time(0.10)
                  self.qtgui_freq_sink_x_0.set_y_axis(-140, 10)
self.qtgui_freq_sink_x_0.set_y_label('Relative Gain', 'dB')
412
                  self.qtgui_freq_sink_x_0.set_trigger_mode(qtgui.TRIG_MODE_FREE, 0.0, 0, "")
                  self.qtgui_freq_sink_x_0.enable_autoscale(False)
                  self.qtgui_freq_sink_x_0.enable_grid(False)
self.qtgui_freq_sink_x_0.set_fft_average(1.0)
417
                  self.qtgui_freq_sink_x_0.enable_axis_labels(True)
                  self.qtgui_freq_sink_x_0.enable_control_panel(False)
                  if not True:
                    self.qtgui_freq_sink_x_0.disable_legend()
                  if "complex" == "float" or "complex" == "msg float":
                     self.qtgui_freq_sink_x_0.set_plot_pos_half(not True)
425
426
427
                  labels = ['', '', '', '',
                              ··, ··, ··, ··, ··j
                  widths = [1, 1, 1, 1, 1,
                             1, 1, 1, 1, 1]
                  colors = ["blue", "red", "green", "black", "cyan",
                              "magenta", "yellow", "dark red", "dark green", "dark blue"]
                  alphas = [1.0, 1.0, 1.0, 1.0, 1.0,
                            1.0, 1.0, 1.0, 1.0, 1.0]
                  for i in xrange(1):
                       if len(labels[i]) == 0:
                           self.qtgui_freq_sink_x_0.set_line_label(i, "Data {0}".format(i))
                       else:
                          self.qtgui_freq_sink_x_0.set_line_label(i, labels[i])
                       self.qtgui_freq_sink_x_0.set_line_width(i, widths[i])
                       self.qtgui_freq_sink_x_0.set_line_color(i, colors[i])
                       self.qtgui_freq_sink_x_0.set_line_alpha(i, alphas[i])
                  self._qtgui_freq_sink_x_0_win = sip.wrapinstance(self.qtgui_freq_sink_x_0.pyqwidget(), Qt.QWidget)
                  self.top_grid_layout.addWidget(self._qtqui_freq_sink_x_0_win)
self.low_pass_filter_0 = filter.fir_filter_ccf(1, firdes.low_pass(
    1, samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76))
                  self.digital_gmsk_demod_1 = digital.gmsk_demod(
                      samples_per_symbol=2,
                      gain mu=0.175.
                      mu=0.5,
                      omega_relative_limit=0.005,
                      freq_error=0.0,
                      verbose=False,
                      log=False,
                  self.blocks_multiply_const_vxx_1 = blocks.multiply_const_vcc((1, ))
                  self.blocks_file_sink_1 = blocks.file_sink(gr.sizeof_char*1, '/home/b/chat_test2.txt', False)
                  self.blocks_file_sink_1.set_unbuffered(False)
                  self.blks2_packet_decoder_1 = grc_blks2.packet_demod_b(grc_blks2.packet_decoder(
461
                           access code='',
                           threshold=-1,
                           callback=lambda ok, payload: self.blks2_packet_decoder_1.recv_pkt(ok, payload),
```

464).	
465)	
466			
467			
468			
469			
470		# Connections	
471 (472		<pre>self.connect((self.blks2_packet_decoder 1, 0), (self.blocks_file_sink_1, 0))</pre>	
473		self.connect((self.blocks_multiply_const_vxx_1, 0), (self.low_pass_filte_r0, 0))	
474		self.connect((self.digital gmsk demod 1, 0), (self.blks2 packet decoder 1, 0))	
475		self.connect((self.low pass filter 0, 0), (self.digital gmsk demod 1, 0))	
476		self.connect((self.low_pass_filter_0, 0), (self.qtgui_freq_sink_x_0, 0))	
477 (<pre>self.connect((self.uhd_usrp_source_0, 0), (self.blocks_multiply_const_vxx_1, 0))</pre>	
478			
479		def closeEvent(self, event):	
480		<pre>self.settings = Qt.QSettings("GNU Radio", "chat2")</pre>	
481		<pre>self.settings.setValue("geometry", self.saveGeometry())</pre>	
482 (483		event.accept()	
484 (<pre>def get_samp_rate(self):</pre>	
485 (return self.samp rate	
486			
487 (<pre>def set_samp_rate(self, samp_rate):</pre>	
488		self.samp_rate = samp_rate	
489		<pre>self.uhd_usrp_source_0.set_samp_rate(self.samp_rate)</pre>	
490		<pre>self.qtgui_freq_sink_x_0.set_frequency_range(0, self.samp_rate)</pre>	
491 (<pre>self.low_pass_filter_0.set_taps(firdes.low_pass(1, self.samp_rate, 200000, 50000, firdes.WIN_HAMMING, 6.76))</pre>	
492			
493	1.0		
494 495	der	main (options=None):	
496		from distutils.version import StrictVersion	
497		<pre>if StrictVersion(Qt.qVersion()) >= StrictVersion("4.5.0"):</pre>	
498		<pre>style = gr.prefs().get string('gtgui', 'style', 'raster')</pre>	
499 (Qt.QApplication.setGraphicsSystem(style)	
500		<pre>qapp = Qt.QApplication(sys.argv)</pre>	
501			
502		data_csv = open('/home/b/20161005_140846.csv', 'r')	
503		<u>csyreader</u> = csy.reader(data_csy)	
504		header = next(csvreader)	
505 506		data_rows = [] for row in csvreader:	
507		data_row.append(row)	
508		data_csv.close()	
509			
510		<pre>for x in range(0,14):</pre>	
511		<pre>out = open('file' + str(x+1) + '.txt', 'w')</pre>	
512		for y in range(0, 100):	
513		out.write(data_rows[y][x+6]+' '+str(x+1)+'0'+str(y+1)+'\n')	
514		out.close()	
515 516		tb = chat2()	
517		LD = Chat2() th.start()	
518		Print "Ready to start"	
519		while True:	
520		<pre>f = open('/home/b/chat_test1.txt', 'r')</pre>	
521		f.flush()	
522		f.readline()	
523		<pre>check = f.readline()</pre>	
524		if (check !=):	
525 526		print_"START"	
526 527		time.sleep(1) break	
527		break tb.stop()	
529			
530		<pre>weight_file = open('/home/b/chat_test1.txt', 'r')</pre>	
531		weight_file.flush()	
532		weight_file.readline()	
533		weights = weight_file.readline()	
534		<pre>w_list = weights.split(",")</pre>	
535		w list.pop()	
536		for m in range(0, len(w list)):	
537		<pre>w list[m] = int(w list[m])</pre>	

538		
539	sensors = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N']	
540		
541	global i, al, bl, cl, dl, el, fl, gl, hl, il, jl, kl, ll, ml, nl	
542	$\mathbf{i} = 0$	
543	a1 = 0	
544	b1 = 0	
545	c1 = 0 d1 = 0	
546		
547	e1 = 0 f1 = 0	
548 549	$g_1 = 0$	
550	$g_1 = 0$ h1 = 0	
551	i1 = 0	
552	j1 = 0	
553	k1 = 0	
554	11 = 0	
555	m1 = 0	
556	n1 = 0	
557		
558	global data0, data1, data2, data3, data4, data5, data6, data7, data8, data9, data10, data11, data12, data13	
560	<pre>read1 = open('/home/b/file1.txt', 'r')</pre>	
561	<pre>readl = open('/home/b/illei.txt', 'F') data0 = readl.readlines()</pre>	
562	length = len(data0)	
563	read1.close()	
564		
565	<pre>read2 = open('/home/b/file2.txt', 'r')</pre>	
566	<pre>data1 = read2.readlines()</pre>	
567	length2 = len(data1)	
568	read2.close()	
569 570	<pre>read3 = open('/home/b/file3.txt', 'r')</pre>	
570	<pre>reads = open('/home/b/illes.txt', 'r') data2 = reads.readlines() </pre>	
572	length = len(data2)	
573	read3.close()	
574		
575	<pre>read4 = open('/home/b/file4.txt', 'r')</pre>	
576	<pre>data3 = read4.readlines()</pre>	
577	length4 = len(data3)	
578 579	read4.close()	
580	<pre>read5 = open('/home/b/file5.txt', 'r')</pre>	
581	data4 = read5.readlines()	
582	length5 = len(data4)	
583	read5.close()	
584		
585	<pre>read6 = open('/home/b/file6.txt', 'r')</pre>	
586	data5 = read6.readlines()	
587 588	<pre>length6 = len(data5) read6.close()</pre>	
589		
590	<pre>read7 = open('/home/b/file7.txt', 'r')</pre>	
591	<pre>data6 = read7.readlines()</pre>	
592	length7 = len(data6)	
593	read7.close()	
594		
595	<pre>read8 = open('/home/b/file8.txt', 'r') here = open('/home/b/file8.txt', 'r')</pre>	
596	<pre>data7 = read8.readlines() length8 = len(data7)</pre>	
597 598	length8 = len(data/) read0.close()	
599		
600	<pre>read9 = open('/home/b/file9.txt', 'r')</pre>	
601	data8 = read9.readlines()	
602	length9 = len(data8)	
603	read9.close()	
604		
605	<pre>read10 = open('/home/b/file10.txt', 'r')</pre>	
606	data9 = read10.read1ines()	
607	length10 = len(data9) read10 = len(data9)	
608 609	read10.close()	
610	<pre>read11 = open('/home/b/file11.txt', 'r')</pre>	
611	data10 = read11.read1mes()	

612	<pre>length11 = len(data10)</pre>	
613	read11.close()	
614		
615	<pre>read12 = open('/home/b/file12.txt', 'r')</pre>	
616	<pre>data11 = read12.readlines()</pre>	
617	<pre>length12 = len(data11)</pre>	
618	read12.close()	
619		
620	<pre>read13 = open('/home/b/file13.txt', 'r')</pre>	
621	data12 = real3.real3.real(ins())	
622	length13 = len(data12)	
623	read13.close()	
624		
625	<pre>read14 = open('/home/b/file14.txt', 'r')</pre>	
626	data13 = read14.readlines()	
627	length14 = len(data13)	
628	read14.close()	
629		
630	<pre>data = list(zip(sensors, w_list))</pre>	
631	sched = WRRScheduler(data)	
632		
633	global a_tot, b_tot, c_tot, d_tot, e_tot, f_tot, g_tot, h_tot, i_tot, j_tot, k_tot, l_tot, m_tot, n_tot, switch	
634	a tot = 0	
635	b tot = 0	
636	c tot = 0	
637	d tot = 0	
638	$a_{\rm corr} = 0$	
639	$f_{tot} = 0$	
640		
641	$g_{tot} = 0$ $h_{tot} = 0$	
642	$i_{tot} = 0$	
643	j_tot = 0	
644	$k_{tot} = 0$	
645	$l_{tot} = 0$	
646	m_tot = 0	
647	n_tot = 0	
648		
649 650	packet_num = 0 iterations = 150	
651	result = []	
652 653	<pre>while (packet num < iterations):</pre>	
654	<pre>choose = sched.get_next() if choose[0] == 'A':</pre>	
655	$a_{\text{tot}} + 1$	
656	if (int	
657	result.append(choose[0])	
658	packet_num += 1	
659	elif choose[0] == 'B':	
660	b_tot += 1	
661	<pre>if (b_tot <= length2):</pre>	
662	result.append(choose[0])	
663	<pre>packet_num += 1</pre>	
664	elif choose[0] == 'C':	
665	c_tot += 1	
666	if (c_tot <= length3):	
667	result.append(choose[0])	
668	packet_num += 1	
669	elif choose[0] == 'D':	
670	d_tot += 1	
671	if (d_tot <= length4):	
672	result.append(choose[0])	
673	packet_num += 1	
674	elif choose[0] == 'E':	
675	e tot += 1	
676	if (e_tot <= length5):	
677	result.append(choose[0])	
678	a packet num += 1	
679	elif chose[0] = 'P':	
680	$f_{\text{tot}} = 1$	
681	if (f_tot <= length6):	
682	result.append(choose[0])	
683	<pre>packet_num += 1</pre>	
684	<pre>> packet_num → i > elif chose[0] = 'G':</pre>	
002		
685	g tot += 1	
685	g_tot += 1	

686	<pre>if (g_tot <= length?):</pre>	
687	result.append(choose[0])	
688	<pre>packet_num += 1</pre>	
689	elif choose[0] == 'H':	
690	$h_{tot} \neq 1$	
691	if (h_tot <= length8):	
692	result.append(choose[0])	
693	<pre>packet_num += 1</pre>	
694	elif choose[0] == 'I':	
695	i_tot += 1	
696	if (i_tot <= length9):	
697	result.append(choose[0])	
698	packet_num += 1	
699	elif choose[0] == 'J':	
700	j_tot += 1	
701	<pre>if (j_tot <= length10):</pre>	
702	result.append(choose[0])	
703	packet_num += 1	
704	elif choose[0] == 'K':	
705	$k_{tot} \neq 1$	
706	<pre>if (k_tot <= length11):</pre>	
707	result.append(choose[0])	
708	packet_num += 1	
709	elif choose[0] == 'L':	
710	1_tot += 1	
711	if (l_tot <= length12):	
712	result.append(choose[0])	
713	packet_num += 1	
714	elif choose[0] == 'M':	
715	$m_{tot} += 1$	
716	if (m_tot <= length13):	
717	result.append(choose[0])	
718	packet_num += 1	
719	elif choose[0] == 'N':	
720	n_tot += 1	
721	<pre>if (n_tot <= length14):</pre>	
722	result.append(choose[0])	
722 723		
722 723 724	result.append(choose[0]) packet_num += 1	
722 723 724 725	<pre>result.append(choose[0]) packet_num += 1 while i < iterations:</pre>	
722 723 724 725 726	<pre>while i < iterations: if result[i] == 'A':</pre>	
722 723 724 725 726 727	<pre>vesult.append(choose[0]) packet_num += 1 vesult(i) == 'A': if result(i) == 'A': switch = 'a'</pre>	
722 723 724 725 726 727 728	<pre>result.append(choose[0]) packet_num += 1 while i < iterations: if result[i] == 'A': switch = 'a' tb = text_tx()</pre>	
722 723 724 725 726 727 728 729	<pre>while i < iterations: if result[i] == 'A': switch = 'a' tb = text_tx() tb.start()</pre>	
722 723 724 725 726 727 728 729 730	<pre>while i < iterations: if result[i] == 'A': switch = 'a' tb = text_tx() tb.start() time.sleep(1)</pre>	
722 723 724 725 726 727 728 729 730 731	<pre>result.append(choose[0]) packet_num += 1 while i < iterations: if result[i] == 'A': switch = 'a' tb = text_tx() tb.start() time.sleep(1) tb.stop()</pre>	
722 723 724 725 726 727 728 729 730 731 732	<pre>while i < iterations: if result[i] == 'A': switch = 'a' tb = text_tx() tb.start() time.sleep(1) tb.stop() i += 1</pre>	
722 723 724 725 726 727 728 729 730 731 732 733	<pre>while i < iterations: if result[i] == 'A': switch = 'a' tb = text_tx() tb.start() tb.start() tb.start() tb.start = 1 a1 += 1</pre>	
722 723 724 725 726 727 728 729 730 731 731 732 733 734	<pre>vesult.append(choose[0]) packet_num += 1 vesult[i] == 'A': switch = 'a' tb = text_tx() tb.start() tb.start() tb.stop() i += 1 al += 1 elif result[i] == 'B':</pre>	
722 723 724 725 726 727 728 729 730 731 731 731 732 733 734 735	<pre>result.append(choose[0])</pre>	
722 723 724 725 726 727 728 729 730 731 732 733 733 734 735 736	<pre>while i < iterations: if result[i] == 'A': switch = 'a' tb = text_tx() tb.start() tb.start() tb.start() tb.stop() i += 1 a1 += 1 ellf result[i] == 'B': switch = 'b' tb = text_tx()</pre>	
722 723 724 725 726 727 728 729 730 731 732 733 734 735 735 736 737	<pre>while i < iterations: if result[i] == 'A': switch = 'a' tb = text_tx() tb.start() tb.start() tb.start() tb.stop() i += 1 al += 1 ellf result[i] == 'B': switch = 'b' tb = text_tx() tb.start()</pre>	
722 723 724 725 726 727 728 729 730 731 731 732 733 734 735 736 737 737 738	<pre>result.append(choose[0])</pre>	
722 723 724 725 726 727 728 729 730 731 732 733 734 735 736 737 738 738 739	<pre>result.append(choose[0]) packet_num += 1 while i < iterations: if result[i] == 'A': switch = 'a' tb = text_tx() tb.start() time.sleep(1) t.stop() i += 1 al += 1 elif result[i] == 'B': switch = 'b' tb = text_tx() tb.start() time.sleep(1) tb.stop() </pre>	
722 723 725 725 726 727 728 730 731 732 733 734 735 734 735 736 737 738 739 739 739 739	<pre>vhile i < iterations: if result.append(choose[0]) packet_num += 1 while i < iterations: if result[i] == 'A': switch = 'a' tb = text_tx() tb.start() time.sleep(1) tb.stop() i += 1 al += 1 ellf result[i] == 'B': switch = 'b' tb = text_tx() tb.start() subscheller() tb.start()</pre>	
722 723 724 725 726 727 728 729 730 731 732 733 733 733 734 735 736 737 738 739 740 741	<pre>result.append(choose[0])</pre>	
722 723 724 725 726 727 729 730 731 732 733 734 735 736 737 738 737 738 739 740 741 742	<pre>result.append(choose[0]) packet_num += 1 while i < iterations: if result[i] == 'A': switch = 'a' tb = text_tx() tb.start() tb.start() tb.start() tb.start() tb.stop() i += 1 a1 += 1 elif result[i] == 'B': switch = 'b' tb = text_tx() tb.start() tb.stop() i += 1 bl += 1 bl += 1 elif result[i] == 'C': </pre>	
722 723 724 725 726 727 728 729 730 731 731 732 733 734 735 735 735 735 735 735 735 735 736 739 740 741 742 742	<pre>result.append(choose[0]) packet_num += 1 while i < iterations: if result[i] == 'A': switch = 'a' tb = text_tx() tb.start() time.sleep(1) tb.stop() i += 1 al += 1 elif result[i] == 'B': switch = 'b' tb = text_tx() tb.start() time.sleep(1) tb.stop() i += 1 bl += i bl += i bl += i switch = 'o' suitch = 'o' switch = 'o'</pre>	
722 723 724 725 726 727 728 729 730 731 732 733 734 735 736 737 737 738 739 740 741 741 743 743 743	<pre>result.append(choose[0])</pre>	
722 723 724 725 726 727 730 730 731 732 733 734 735 736 735 736 737 738 738 739 740 741 742 742 744 745	<pre>result.append(choose[0])</pre>	
722 723 724 725 726 727 730 731 731 732 733 734 735 736 735 736 737 738 739 740 741 742 743 744 745 745	<pre>result.append(choose[0])</pre>	
722 723 724 725 726 727 728 729 730 731 732 733 734 734 735 736 737 738 739 737 738 739 740 741 741 742 743 744	<pre>result.append(choose[0]) packet_num += 1 while i < iterations: if result[i] == 'A': switch = 'a' tb = text_tx() tb.start() tb.start() tb.stap() i += 1 al += 1 elif result[i] == 'B': switch = 'b' tb = text_tx() tb.start() tb.start(</pre>	
722 723 724 725 726 727 730 731 732 733 734 735 736 737 738 739 740 741 742 743 744 745 744 745 747 748	<pre>result.append(choose[0]) packet_num += 1 while i < iterations: if result[i] == 'A': switch = 'a' tb = text_tx() tb.start() time.sleep(1) tb.stop() i += 1 elif result[i] == 'B': switch = 'b' tb = text_tx() tb.start() tb.stop() i += 1 bl += 1 bl += 1 elif result[i] == 'C': switch = 'a' tb = text_tx() tb.start() tb.start()</pre>	
722 723 724 725 726 727 730 731 732 733 734 735 736 735 736 737 738 739 740 741 742 743 744 745 746 747 748 748 749	<pre>result.append(choose[0]) packet_num += 1 while i < iterations: if result[i] == 'A': switch = 'a' tb = text_tx() tb.stop() ti.s.stop() i += 1 al += 1 elif result[i] == 'B': switch = 'b' tb = text_tx() tb.stop() i += 1 bl += 1 cl += 1 bl += 1 cl += 1</pre>	
722 723 724 725 726 727 728 730 731 731 732 733 734 735 735 736 735 736 737 738 739 740 741 742 743 744 745 746 747 746 749 749 749 749	<pre>xesult.append(choose(0)) packet_num += 1 while i < iterations: if result[i] == 'A': switch = 'a' tb = text_tx() tb.start() time.sleep(1) tb.stop() i += 1 al += 1 elif result[i] == 'B': switch = 'b' tb = text_tx() tb.start() time.sleep(1) tb.start(i] == 'C': switch = 'c': switch = 'a' tb = text_tx() tb.start() tb.start(i) tb.s</pre>	
722 723 724 725 726 727 730 731 732 733 734 735 736 735 736 737 738 739 740 741 742 743 744 745 746 747 748 748 749	<pre>result.append(chose[0]) packet_num += 1 while i < iterations: if result[i] == 'A': switch = 'a' tb = text_tx() tb.start() time.sleep(1) tb.stap() i += 1 al += 1 elif result[i] == 'B': switch = 'b' tb = text_tx() tb.start() time.sleep(1) tb.start() tb.start()</pre>	
722 723 724 725 726 727 730 731 732 733 733 734 735 736 737 738 739 740 741 742 743 744 744 745 746 747 748 748 749 749 752	<pre>result.append(chose[0]) packet_num += 1 while i < iterations: if result[i] == 'A': switch = 'a' tb = text_tx() tb.start() t</pre>	
722 723 724 725 726 727 730 731 731 732 733 734 735 736 737 735 736 737 738 739 740 741 743 742 743 744 745 745 746 745 749 749 749 749 749 749 749 745 749 749 749 749 749 749 749 749 752 753	<pre>result.speed(choose[0]) packet_num += 1 while i < iterations: if result[i] == 'A': switch = 'a' tb = text_tx() tb.start() tb.start() tb.start() tb.start() tb.start() tb.start() tb = text_tx() tb.start() tb.start()</pre>	
722 723 724 725 726 727 730 731 732 733 734 735 736 737 737 738 739 737 738 739 739 740 741 742 743 744 744 745 744 745 744 745 747 748 747 748 749 747 748 749 745 750 751	<pre>result.append(choose[0]) packet_num += 1 while i < irr(int = 1</pre>	
722 723 724 725 726 727 730 731 732 733 733 734 735 736 737 738 738 739 740 741 742 743 744 745 744 745 744 745 748 749 751 752 753	<pre>result.append(choose[0]) packt_num += 1 while i < iterations: if result[i] == 'A': switch = 'a' tb = text_tx() tb.stop() i += 1 ellf result[i] == 'B': switch = 'b': tb = text_tx() tb.stop() i += 1 bl += 1 clif result[i] == 'C': switch = 'c' tb = text_tx() tb.stop() i += 1 bl += 1 bl</pre>	
722 723 724 725 726 727 730 731 732 733 734 735 736 737 738 736 737 738 739 740 741 743 744 745 744 745 744 745 745 746 747 748 749 755 755 755 755	<pre>result.append(choose[0]) packt_num += 1 while i < iterations: if result[] == 'A': switch = 'a' tb = test_ts() tb.start() tb.start() tb.start() tb.start() tb = test_ts() tb.start() tb.start()</pre>	
722 723 724 725 726 727 730 731 732 733 733 734 735 736 737 738 738 739 740 741 742 743 744 745 744 745 744 745 748 749 751 752 753	<pre>while i - apend (choose [0]) packt_num += 1 while i < iterations: if result[i] == 'A': switch = 'a' tb = text_tx() tb = te</pre>	
722 723 724 725 726 727 730 731 732 733 734 735 736 737 737 738 739 739 739 739 739 739 739 739 739 739	<pre>result.append(choose[0]) packt_num += 1 while i < iterations: if result[] == 'A': switch = 'a' tb = test_ts() tb.start() tb.start() tb.start() tb.start() tb = test_ts() tb.start() tb.start()</pre>	

760		<pre>tb = text_tx()</pre>
761		tb.start()
762		time.sleep(1)
763		tb.stop()
764		i += 1
765		e1 += 1
766		<pre>elif result[i] == 'F':</pre>
767		switch = 'f'
768		tb = text_tx()
769		tb.start()
770		time.sleep(1)
771		tb.stop()
772		i += 1
773		f1 += 1
774		<pre>elif result[i] == 'G':</pre>
775		switch = 'g'
776		tb = text_tx()
777		tb.start()
778		
		time.sleep(1)
779		tb.stop() i += 1
780		
781		g1 += 1
782		<pre>elif result[i] == 'H':</pre>
783		switch = 'h'
784		<pre>tb = text_tx()</pre>
785		tb.start()
786		time.sleep(1)
787		tb.stop()
788		i += 1
789		h1 += 1
790		<pre>elif result[i] == 'I':</pre>
791		switch = 'i'
792		<pre>tb = text_tx()</pre>
793		tb.start()
794		time.sleep(1)
795		tb.stop()
796		i += 1
797		i1 += 1
798 🤅		<pre>elif result[i] == 'J':</pre>
799		switch = 'j'
800		<pre>tb = text_tx()</pre>
801		tb.start()
802		time.sleep(1)
803		tb.stop()
804		i += 1
805		j1 += 1
805 8		<pre>lif result[i] == 'K':</pre>
		switch = 'k'
807		
808		tb = text_tx()
809		tb.start()
810		time.sleep(1)
811		tb.stop()
812		i += 1
813		k1 += 1
814 🤅		<pre>elif result[i] == 'L':</pre>
815		switch = '1'
816		tb = text_tx()
817		tb.start()
818		time.sleep(1)
819		tb.stop()
820		i += 1
821		11 += 1
822 (elif result[i] == 'M':
822 (switch = 'm'
824		<pre>tb = text_tx() tb =text_()</pre>
825		tb.start()
826		time.sleep(1)
827		tb.stop()
		i += 1
828		m1 += 1
829		
		<pre>elif result[i] == 'N':</pre>
829		<pre>elif result[i] == 'N': switch = 'n'</pre>
829 (830 (Ģ	

834	time.sleep(1)				
835	tb.stop()				
836	i += 1				
837 838	nl += 1				
39	else:				
40	print "Bror: Empty Result"				
	tb = ack2()				
41	th.start()				
42	while True:				
43	<pre>ack_file = open('/home/b/chat_test2.txt', 'r')</pre>				
44	ack_file.flush()				
45	ack_file.readline()				
46	<pre>ack0 = ack_file.readline()</pre>				
17	if (ack0 == 'ACK\n'):				
18	print, "RECEIVING ACK"				
19	time.sleep(1)				
50	break				
1	ack_file.close()				
2	time.sleep(1)				
3	tb.stop()				
4	if $(i > 28)$ and $(i \le 20 = 0)$:				
5	tb = chat2()				
6	tb.start()				
7	while True:				
8	<pre>f = open('home/b/chat_testl.txt', 'r') f first</pre>				
59	f.flush()				
	f.readline()				
51	check = f.readline()				
52	if (check l= ""):				
53	time.sleep(1)				
54	break				
65	tb.stop()				
56	<pre>weight_file = open('/home/b/chat_testl.txt', 'r') </pre>				
57	weight_file.flush()				
68	weight_file.readline()				
69	<pre>weights = weight_file.readline()</pre>				
70 71	<pre>w_list = weights.split(",")</pre>				
	w_list.pop()				
72	for m in range(0, len(w list)):				
73	w_list[m] = int(w_list[m])				
74	<pre>data = list(zip(sensors, w_list))</pre>				
75 76	<pre>sched = WRRScheduler(data) count = 0</pre>				
70					
78	new result = []				
	<pre>while (count < (iterations-i)):</pre>				
79	<pre>new_choose = sched.get_next() ff mer_choose() == lot() </pre>				
	if new choose $[0] == 'A'$:				
31 32	$a_{tot} + 1$				
	if (a tot <= length1):				
33	<pre>new result.append(new_choose[0]) coupt t=</pre>				
34	count += 1				
35	elif new_choose[0] == 'B':				
36 37	$b \cot t = 1$				
37	if (b_tot <= length2):				
	new_result.append(new_choose[0])				
9	count += 1				
90 91	elif new_choose[0] == 'C':				
2	c_{\pm} to $t \neq 1$				
	if (c tot <= length3):				
3	<pre>new_result.append(new_choose[0]) count f</pre>				
	count += 1				
5	elif new_choose[0] == 'D':				
6	$d_{tot} \neq 1$				
7	if (d_tot <= length4):				
8	<pre>new_result.append(new_choose[0]) </pre>				
9	count += 1				
00	<pre>elif new_choose[0] == 'E':</pre>				
01	e_tot += 1				
02	<pre>if (e_tot <= length5):</pre>				
03	<pre>new_result.append(new_choose[0])</pre>				
	count += 1				
04	<pre>elif new_choose[0] == 'F':</pre>				
04 05					
	<pre>f_tot += 1 if (f_tot <= length6):</pre>				

908	new_result.append(new_choose[0])					
909	count += 1					
910	<pre>elif new_choose[0] == 'G':</pre>					
911	g_tot += 1					
912	if (g_tot <= length7):					
913	new_result.append(new_choose[0])					
914	count += 1					
915	elif new_choose[0] == 'H':					
	h tot ± 1					
916						
917	<pre>if (h_tot <= length8):</pre>					
918	<pre>new_result.append(new_choose[0])</pre>					
919	count += 1					
920	elif new_choose[0] == 'I':					
921	i_tot += 1					
922	if (i tot <= length9):					
923						
	new_result.append(new_choose[0])					
924	count += 1					
925	elif new_choose[0] == 'J':					
926	j_tot += 1					
927	<pre>if (j_tot <= length10):</pre>					
928	new_result.append(new_choose[0])					
929	count += 1					
930	<pre>elif new_choose[0] == 'K':</pre>					
931	k_tot += 1					
932	<pre>if (k_tot <= length11):</pre>					
933	new_result.append(new_choose[0])					
934	count += 1					
935	elif new_choose[0] == 'L':					
936	l_tot += 1					
937	<pre>if (1_tot <= length12):</pre>					
938	new result.append(new choose[0])					
939	count += 1					
940	<pre>elif new_choose[0] == 'M':</pre>					
941	m tot += 1					
942	if (m_tot <= length13):					
943	new_result.append(new_choose[0])					
944	count += 1					
945	elif new_choose[0] == 'N':					
946	$n \pm tot += 1$					
947	if (n tot <= length14):					
948						
	new_result.append(new_choose[0])					
949	count += 1					
950	<pre>if (len(new_result) < iterations):</pre>					
951	<pre>for v in range(iterations-len(new_result)):</pre>					
952	new_result.insert(0,0)					
953	result = new_result					
954	choose = new_choose					
955						
956	switch = 'eof'					
957	tb = text_tx()					
958	tb.start()					
959	time.sleep(1)					
960	tb.stop()					
961						
962	print('done')					
963	exit()					
964						
965	def quitting():					
966	tb.stop()					
967	tb.wait()					
968	<pre>qapp.connect(qapp, Qt.SIGNAL("aboutToQuit()"), quitting)</pre>					
969	<pre>qapp.exec_()</pre>					
970						
971						
972 🕨	if == 'main':					
973	main()					
974						

Figure A.4: Smart Weighted Round-Robin Algorithm for Field Sensor Radio