Ted Krimmer

Ripper Ranger plan is to create a way for the messages captured by the ranger from the Remus 100 to be sent to an Arduino computer to record and use.

Ripper Ranger

(Working title) Project Plan

# Project Plan - Ripper Ranger Serial to printer

Ted Krimmer – 12/01/2002

Contents

[Project Plan - Ripper Ranger Serial to printer 1](#_Toc124776436)

[1. Introduction 2](#_Toc124776437)

[2. Project ideas 2](#_Toc124776438)

[2.2. Ripper Ranger Screen to text printing 2](#_Toc124776439)

[2.3. Ripper Ranger Serial to printer 3](#_Toc124776440)

[3. Project Plan 3](#_Toc124776441)

[3.1. Ripper Ranger Screen to text printing Plan 3](#_Toc124776442)

[3.2. Ripper Ranger Serial to printer Plan 4](#_Toc124776443)

[3.2.1. requirements 4](#_Toc124776444)

[3.2.2. Steps 5](#_Toc124776445)

[3.2.2.3. Arduino Printer 6](#_Toc124776446)

[3.2.2.4. Arduino Serial Passthrough 6](#_Toc124776447)

[3.2.2.5. Additional nice-to-have components 6](#_Toc124776448)

[3.2.2.6. Arduino Basic code flow 6](#_Toc124776449)

[4. Conclusion 7](#_Toc124776450)

[Appendix 8](#_Toc124776451)

# Introduction

This document shows the planning of the implementation for the project named “Ripper Ranger”. The basic plan for the Ripper Ranger is to create a way for the messages captured by the ranger from the Remus 100 to be automatically recorded. Best case the messages are sent to an Arduino computer, recorded, and printed, and passed to the mission laptop simultaneously.

Currently, the message from the ranger is passed to the mission laptop. This message is programmed to be sent from the Remus to the ranger every minute. However, it can occasionally be missed due to the nature of underwater communication methods. The issue with this system is that it only displays the latest message and doesn’t keep track of previous messages. This causes operators to spend significant time watching for new messages and handwriting this into a logbook. This is done to ensure that if the operators lose communication with the AUV (Autonomous underwater vehicle) and need to find and recover the vehicle, they will have the last few know location from the messages to determine a starting search around.

The project we have decided to implement is to display and record these messages allowing the operators more freedom to focus on the mission. Ultimately, the project’s initial goal is to remove the need for operators to check and write down each message manually.

# Project ideas

There are two ideas for how to implement this. These are both described in “Appendix A – Internship plan” titled “Ripper Ranger Screen to text printing” and “Ripper Ranger Serial to printer.”

## Ripper Ranger Screen to text printing

The first obstacle is that the message is sent as a serial message which is then read by the mission laptop. This is a hexadecimal-like message with no documentation on how it is read. This first plan is to ignore the message and write a program to print the text seen on the screen. While this mainly eliminates the need for operators to log messages, this solution only works if the laptop works. If the laptop is unusable, the operators still need to take notes from the ranger.

This can be broken down into two stages. The first stage is to take an image from the message box on the VIP (Vehicle Interface Program), the program used on the mission laptop. Then take that image and record the text on the image to a text document. The second stage is to send that text to a printer.

This project is a good way of prototyping the idea and doesn’t rely on solving the issue with the hexadecimal-like message.

## Ripper Ranger Serial microcontroller

The plan for this project is to intercept the ranger’s messages leaving the ranger. Then use an Arduino board or similar device to capture the message, deciphers it, sends it to a printer then pass the message to the mission laptop. This will only be possible If we can decipher the hexadecimal-like message from the ranger.

This plan could also have things added to it in the future, such as GPS, LCD screen, time tracking, message saving and file storage.

# Project Plan

## Ripper Ranger Screen to text printing Plan

Starting out with the screen recording idea. The plan is to use write python script that the user can run in the background on the mission laptop. Using the “pyscreenshot” library, we can get a screenshot of just the message box of VIP. Then using the “pytesseract” python library, we can transcribe the image to text. Then save that text to a file in two locations. One file will be a log with a timestamp. The another will be a temporary file that is sent to a printer.

The idea is for this to be a very basic proof of concept. The flow of how the program will work is displayed in the diagram below. Image 1 is the basic flow with details about how the code will run.

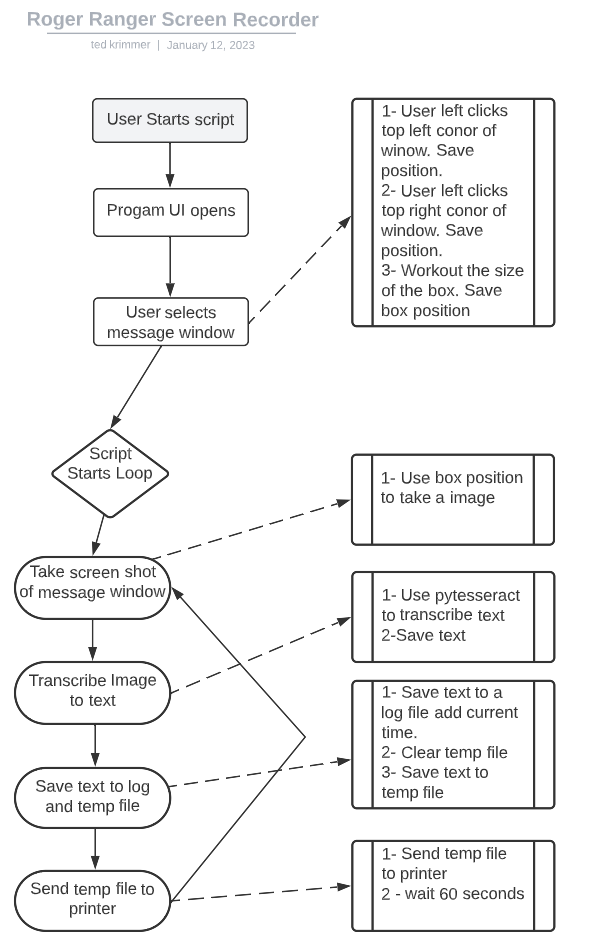


Figure - Basic Flow

## Ripper Ranger Serial microcontroller

### requirements

This project requires both hardware and software. The hardware required is an Arduino microcontroller (Or equivalent), This is used to store the software and run the other hardware components. The price varies but it is roughly $50. We will also need a way to read the serial messages from the rover. Also, need a way for the passthrough to send the messages to the laptop. This can be done straight from the board; however, we recommend adding two RS232 to Serial Converter so we can use standard cables. These can be purchased for about $8 each.

We will also need a printer. We recommend using the [Mini Thermal Receipt Printer](https://www.adafruit.com/product/597) from adafruit. These can be purchased from the adafruit website for $49.95. Thermal printers would be ideal because they are small and don’t require ink.

The printer will need a separate power supply. The Arduino board outputs 5 volts at a max of 0.04 amps. This printer requires 5/9 volts at 2 amps, so it cannot be power directly by the Arduino. There are many different options for power. A portable power pack could be used. We found the ITECH10KPD power bank for $40 + shipping. We have not personally used this; however, it should be an excellent portable option based on the specifications; something similar could also be used. Other options are using a wall plug or potentially using a USB cable from the laptop. In development, we will use the USB from a laptop if it works.

A total estimate of the basic options will be between $120-$150 AUS without the portable power bank.

For additional options – Prices are a rough estimate depending on choice.

GPS unit - $40 (duinotech neo-7m).

LCD or LED screen - $20.

SD card reader - $15

64GB SD card - $15

The total price estimate for the essentials plus additions is around $210-$250 AUS.

### 3.2.2. Steps

##### 3.2.2.1. Putty

First up, we need to decipher the messages sent from the AUV. Currently, we can see how these messages are received by the laptop using a program called putty. The image (Figure 2) below shows the messages. The entire document can be seen in Appendix B.

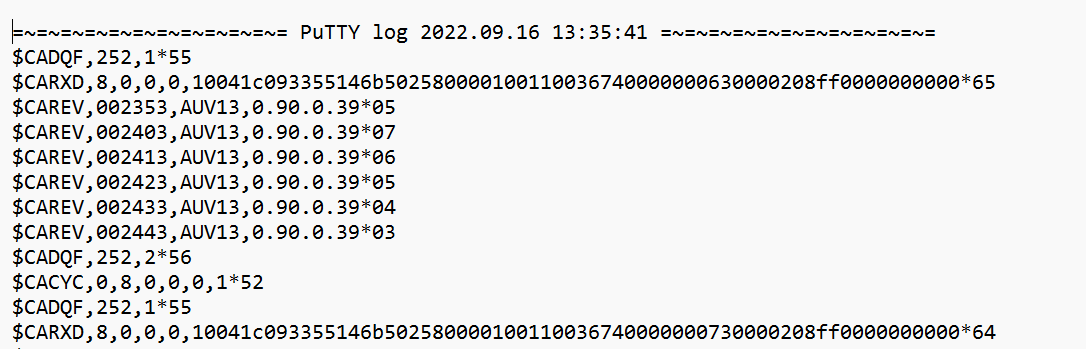


Figure - Putty Log

I plan to go out with the team on a Remus mission and record both the messages as above and take a photo of what the ranger displays. Using these images, we can try and figure out how it is translated.

##### 3.2.2.2. Arduino Serial reader

Once we have figured out how the messages are translated ill will be able to start work on the Arduino board. First part is to get the Arduino to capture serial messages. To start we will manually send messages from my laptop.

Next part is to write a function that will capture the relevant message, save it, and then format it. We may need to filter out certain types of messages which are not required.

### 3.2.2.3. Arduino Printer

After getting the message, we will need to connect the printer and send the information to be printed. This stage might have a bit of trial and error to make the message look nice. Adafruit has great libraries and documentation to use.

### 3.2.2.4. Arduino Serial Passthrough

Finally, we need to send the message out to the laptop. The idea is that this project shouldn’t affect how messages are viewed. Making the integration as seamless as possible.

### 3.2.2.5. Additional nice-to-have components

After the basic functions have been achieved, the system can be expanded. Depending on the budget and time constraints, GPS, LCD screen, time tracking, message saving, file storage and much more can be added.

GPS could track the location of the boat. A functional implementation could be tracking the boat in at one-minute intervals. It could be interesting for operators to see where they are compared to the AUV if they miss a few ranger messages in a row. This will give the team a better understanding of how they need to position the boat for more reliable communication with the Remus.

LCD Screen could be added to display basic information. This could include the last ranger mission as a backup, estimated time till the next ranger message and more.

The great thing about this project is that it can be easily customised to add hundreds of different components and functions.

### 3.2.2.6. Arduino Basic code flow

Below is a diagram of how the basic Arduino code will run. It will start as soon as it turns on. It will start tracking time and then start waiting for serial messages. Once it receives the new message, it will check to see if it’s relevant. If it is, it will decipher, format, and print the message and then wait for a new message. It will use the serial passthrough to send the original message to the laptop. If the message isn’t relevant, it will delete it and continue waiting for messages.

##### 

Figure - Arduino basic code flow

# Conclusion

Successful implementation of the basic idea of this project will help operator with every mission. It will give another back up options if the laptop stops working and free up the team to focus on the mission and safety or the team and AUV. In addition, it will help keep better records for mission and eliminate the need to hand write messages into the logbook. The number of paths this project can take with the additional components are very interesting and could have more benefits in research, training and operators’ mission awareness. Please see Appendix C and Appendix D to view reports of the project’s development and the final project.

# Appendix

|  |  |
| --- | --- |
| Appendix A |  |
| Appendix B |  |
| Appendix C |  |
| Appendix D |  |