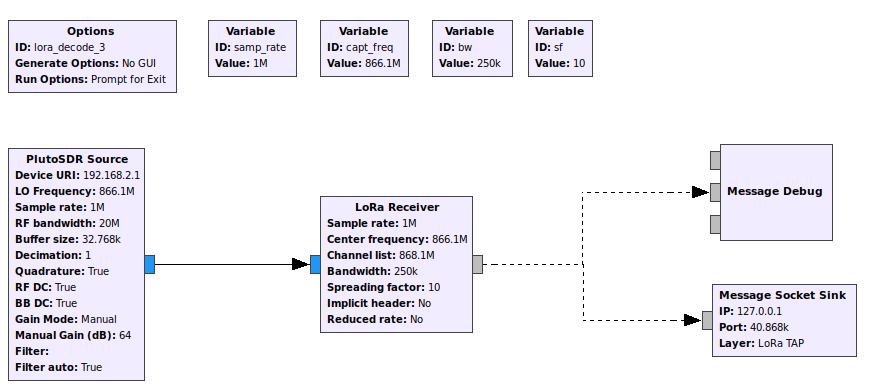
# **Cap**t**ure & Decode LoRa signals**

Η εργασία πραγματοποιήθηκε σε δυο στάδια, το πρώτο ήταν η δημιουργία του flow graph ενός LoRa Receiver με χρήση του GNU Radio Companion και το δεύτερο η επεξεργασία και η προσθήκη κώδικα στο αρχείο python που δημιουργήθηκε από το GNU Radio Companion, προκειμένου να επεκταθεί η λειτουργικότητα του.

Στο πρώτο στάδιο χρησιμοποιήθηκαν, τα modules του κατασκευαστή του Pluto SDR (PlutoSDR Source) και του χρήστη rpp0 (<https://github.com/rpp0/gr-lora/tree/gr3.7>) για τη λήψη των LoRa σημάτων (LoRa Receiver).

Το παραχθέν flow graph φαίνεται παρακάτω και βρίσκεται στο αρχείο lora\_decode.grc:



Τα modules που χρησιμοποιήθηκαν είναι τα εξής:

* PlutoSDR Source: εκπροσωπεί την φυσική συσκευή SDR που χρησιμοποιήθηκε για την λήψη των μηνυμάτων, σε αυτό ρυθμίζεται ο δίαυλος επικοινωνίας με την συσκευή καθώς και οι ρυθμίσεις της συσκευής (Sample Rate, Capture Frequency, Bandwidth)
* LoRa Receiver: είναι υπεύθυνο για την σωστή λήψη των LoRa σημάτων, σε αυτό γίνονται οι αλλαγές των συνθηκών λήψης όπως bandwidth και spreading factor.
* Message Debug: χρησιμοποιήθηκε για την αποθήκευση των ληφθέντων μηνυμάτων.
* Message Socket Sink: χρησιμεύει για την προώθηση των LoRa σημάτων που λαμβάνει το SDR, σαν UDP μηνύματα μέσω της loopback IP (127.0.0.1), στο Wireshark.

Στο δεύτερο στάδιο έπρεπε να τροποποιηθεί το αρχικό lora\_decode.py κατάλληλα ώστε να μπορεί να εναλλάσσει κυκλικά τις ρυθμίσεις του bandwidth και του spreading factor, μέχρι να εντοπίσει τις αντίστοιχες με τον πομπό ώστε να καταστεί εφικτή η λήψη των μηνυμάτων. Σε περίπτωση που αυτές αλλάξουν, θα πρέπει να είναι σε θέση να προσαρμοστεί ώστε να συνεχιστεί αδιάκοπα η λήψη.

Τέλος με την εκτέλεση του αρχείου lora\_decode\_final.py και την παρακολούθηση της loopback διεύθυνσης μέσω του Wireshark, έχουμε το επιθυμητό αποτέλεσμα, δηλαδή τη λήψη και αποκωδικοποίηση LoRa μηνυμάτων, με μοναδικό πρόβλημα τη μη λήψη μηνυμάτων σε bandwidth 125kHz.