Руководство пользователя по запуску программного обеспечения автопилота в режиме SITL с пакетом MATLAB Simulink.

**Содержание**

[Введение](#_wr9voysq0lg) 2

[1. Перечень необходимого предустановленного программного обеспечения](#_l4jdbhcp3jxh) 3

[2. Установка PX4 и multirotor simulator jMAVSim под Linux](#_dog63ylm9a0h) 4

[3. Установка компилятора](#_lg7rn52fls71) 5

[4. Запуск SITL в режиме работы с Simulink](#_z96g4xqft5al) 6

[4.1 Подготовка среды Simulink для передачи данных по mavlink](#_zbwu8itfpmp8) 7

[4.1.1 Компиляция s-function и создании mavlink\_bus\_msg](#_kkoryasafryy) 7

[4.1.2 Построение модели сообщения в Simulink](#_14xpqjpiaj9h) 9

[4.1.2.1 Модель передаваемого сообщения по mavlink](#_3u2ofgjqbu88) 9

[4.1.2.2 Модель принимаемого сообщения по mavlink](#_8p2q0ib0qj35) 12

[4.1.3 Настройка симуляции и первый запуск модели](#_mfyd82w8u7rc) 16

[4.2 Обмен heartbeat между SITL и Simulink](#_qc84knw6fewk) 20

# Введение

Программное обеспечение автопилота PX4 имеет возможность работы в режиме SITL. Этот режим означает, что PX4 будет работать в зацикленном режиме и позволит осуществлять симуляцию взаимодействия управляющего программного кода автопилота с математической моделью беспилотных летательных аппаратов (далее БПЛА). Simulink, являющийся одним из пакетов среды MATLAB, хорошо приспособлен для имитационного моделирования, что позволяет осуществлять построение сложных динамических систем (частным случаем является математическая модель летательного аппарата) и их анализ. Доработка режима SITL, в котором возможно будет реализовать обмен между средой MATLAB и автопилотом, позволит произвести симуляцию работы программного обеспечения PX4 с любой математической моделью БПЛА, построенной в графической среде simulink.

# 1. Перечень необходимого предустановленного программного обеспечения

Для осуществления симуляции автопилота с пакетом simulink на вычислительном устройстве должны иметься:

* исходный код PX4;
* установленный Matlab;
* компиляторы gcc версиями:

1. 5.4 и выше;
2. 4.9.4.

Рекомендуется установить:

* анализатор сетевого трафика Wireshark, для возможности отслеживания передачи пакетов по UDP.
* в качестве операционной системы использовать дистрибутив Ubuntu Linux LTS (16.04), поскольку на этой версии системы отлаживалась работа автопилота с симулятором и его запуск в режиме SITL с Simulink.

# 2. Установка PX4 и multirotor simulator jMAVSim под Linux

Перед началом работы необходимо перенести исходники PX4 на свой ПК. Для этого необходимо склонировать репозиторий находящийся по [ссылке](https://github.com/katofalka/Firmware.git) на свой ПК. Для этого в терминал необходимо ввести:

git clone <https://github.com/katofalka/Firmware.git>

Дождаться окончания загрузки. После чего перейти в папку склонированного репозитория. В терминале необходимо набрать команду:

cd ~/Firmware

Далее нужно перейти на ветку sitl\_matlab, для этого набрать в терминале команду:

git checkout sitl\_matlab

Необходимо обновить подмодули проекта, введя в терминал команду:

git submodule update --init --recursive

По окончанию обновления подмодулей, выполнить сборку прошивки автопилота:

make posix\_sitl\_default jmavsim

При успешной сборке в консоль должно вывестись сообщение “BUILD SUCCESSFUL”(см. рис. 1)

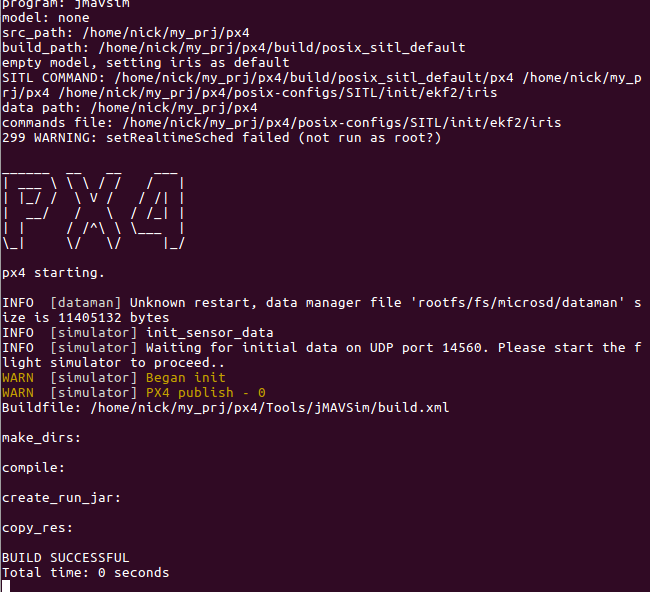


Рисунок 1. Результат успешной сборки ПО автопилота

В случае успеха симуляции следует перейти к следующему этапу руководства “Установка компилятора”.

# 3. Установка компилятора

Для компиляции s-function simulink необходим компилятор gcc версии 4.9.x.

Если в операционной системе нет такого компилятора, его необходимо скачать. Для этого последовательно введите в терминал linux команды и дождитесь окончания загрузки:

sudo apt install gcc-snapshot;

sudo apt install gcc-4.9 g++-4.9

Текущую версию компилятора можно узнать, введя в терминал:

gcc -v

Далее необходимо задать компилятору 4.9.х наивысший относительно других gcc в системе приоритет. Для этого командой update-alternatives изменяются приоритеты присутствующих в системе компиляторов (см. рис. 3):

sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-y xx --slave /usr/bin/g++ g++ /usr/bin/g++-y ,

где хх - значение приоритета, лежащее в числовом диапазоне [0;60];

y - версия компилятора.



Рисунок 3. Пример изменения версии компилятора, используемого по умолчанию ОС

После этого необходимо повторно проверить версию компилятора, если активна другая версия необходимо понизить значение ее приоритета.

# 4. Запуск SITL в режиме работы с Simulink

## 

## 4.1 Подготовка среды Simulink для передачи данных по mavlink

Для передачи данных по протоколу mavlink необходимо создать шины simulink, s-function и построить саму модель. Создание шин и s-function происходит при помощи скриптов matlab, находящихся в папке simulink\_mavlink по [ссылке](https://drive.google.com/drive/folders/1yIhblNpFXVdKWqI7AeETc4SwknmqriYM?usp=sharing). Созданные при помощи них функции и шины позволяют осуществить кодирование/декодирование сообщений по протоколу mavlink и передать их по UDP.

### 4.1.1 Компиляция s-function и создании mavlink\_bus\_msg

Скачайте папку “Simulink” в свой локальный репозиторий перейдя по [ссылке](https://drive.google.com/open?id=1ujNdOiW3Rsllbzr1EwmGrpvAh3lZjCqu). Запустите MATLAB, добавьте скачанную папку с подкаталогами и опционально другие необходимые папки в рабочую область.(см. рис. 5).

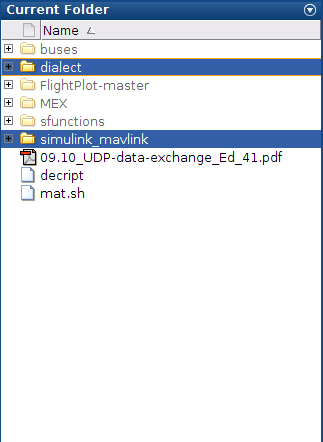


Рисунок 5. Состояние рабочей области MATLAB

Создание шины и s-function рассматривается на примере с сообщением mavlink\_msg\_heartbeat. Для этого в командном окне MATLAB необходимо вызвать функцию кодирования сообщения (см. рис. 6) - create\_sfun\_encode. Функция имеет три входных аргумента: полный путь к сообщению mavlink , значение System\_ID и значение Component\_ID.

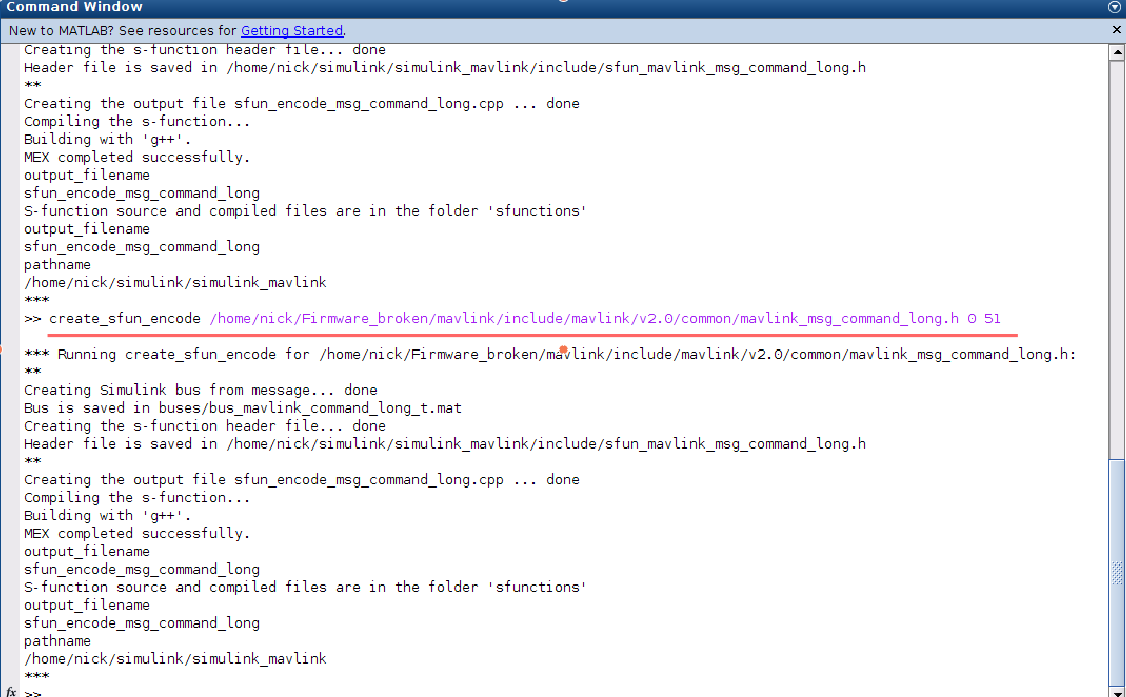


Рисунок 6. Результат вызова функции кодирования сообщения

Полный путь к сообщению является путем к заголовочному файлу сообщения, используемого диалекта mavlink. В настоящей реализации используются сообщения из общего набора “common”, находящемуся по пути: /simulink/simulink\_mavlink/include/mavlink/v2.0/common/<имя заголовочного файла>.h.

В качестве параметров System\_ID передается 0 а в качестве Component\_ID передается 51. При успешном завершении работы скрипта будет создана шина (см. рис. 7), а в папке ~/simulink/simulink\_mavlink/sfunctions будут находиться сгенерированный и откомпилированный файлы функции.

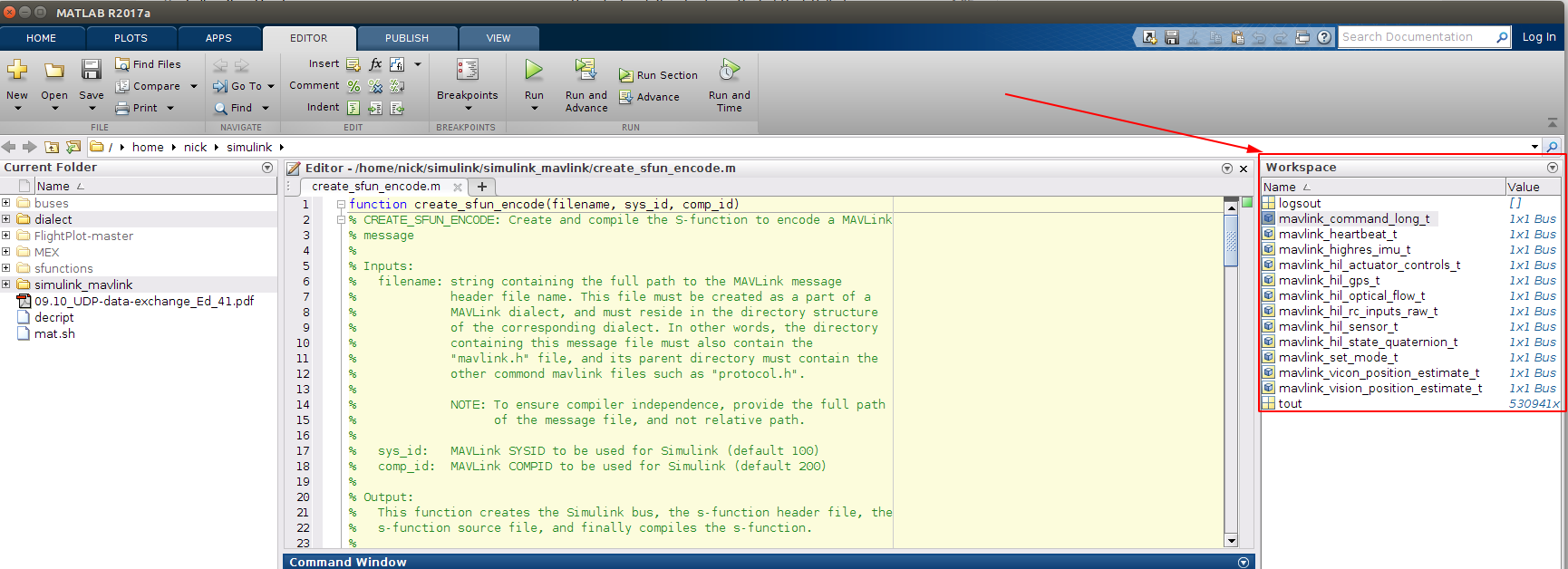


Рисунок 7. Наличие созданных шин в рабочей области MATLAB

**Важно**! Необходимо сохранить рабочее пространство во избежании повторения цикла подготовки.

### 4.1.2 Построение модели сообщения в Simulink

#### 4.1.2.1 Модель передаваемого сообщения по mavlink

Необходимо запустить пакет Simulink. В рабочей области разместить компоненты, как показано на рисунке 8.

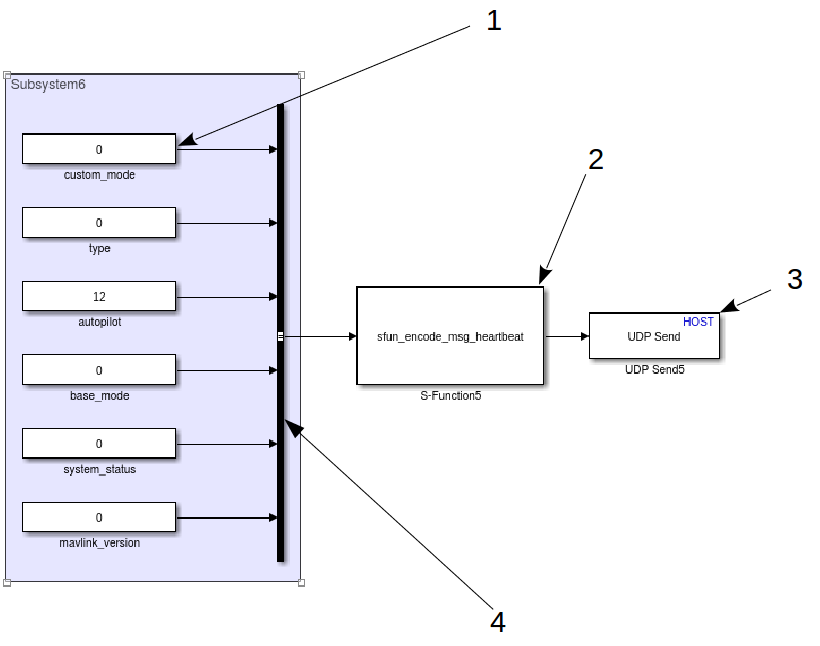


Рисунок 8. Модель сообщения mavlink\_heartbeat. 1 - блок Constant Value; 2 - блок S-Function; 3 - блок UDP Send из группы Embedded Coder; 4 - блок Bus Creator.

Необходимо указать тип данных, используемых блоком Constant. Для этого двойным нажатием ЛКМ раскрывается меню параметров блока, далее произвести переход на вкладку “Signal Attributes”, после чего нужно из всплывающего контекстного меню выбрать тип данных поля “Output data type” (см. рис. 9).

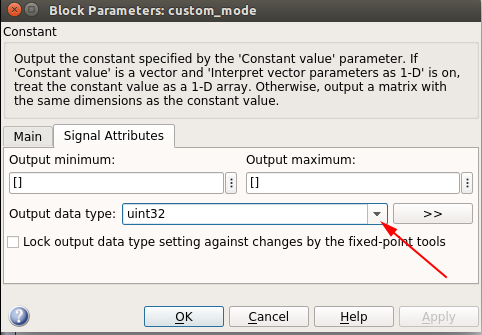


Рисунок 9. Окно параметров блока Constant Values

Чтобы правильно задать размер поля, необходимо вызвать окно управления шиной. В данном окне можно посмотреть порядок следования полей и их размер (см. рис. 10).

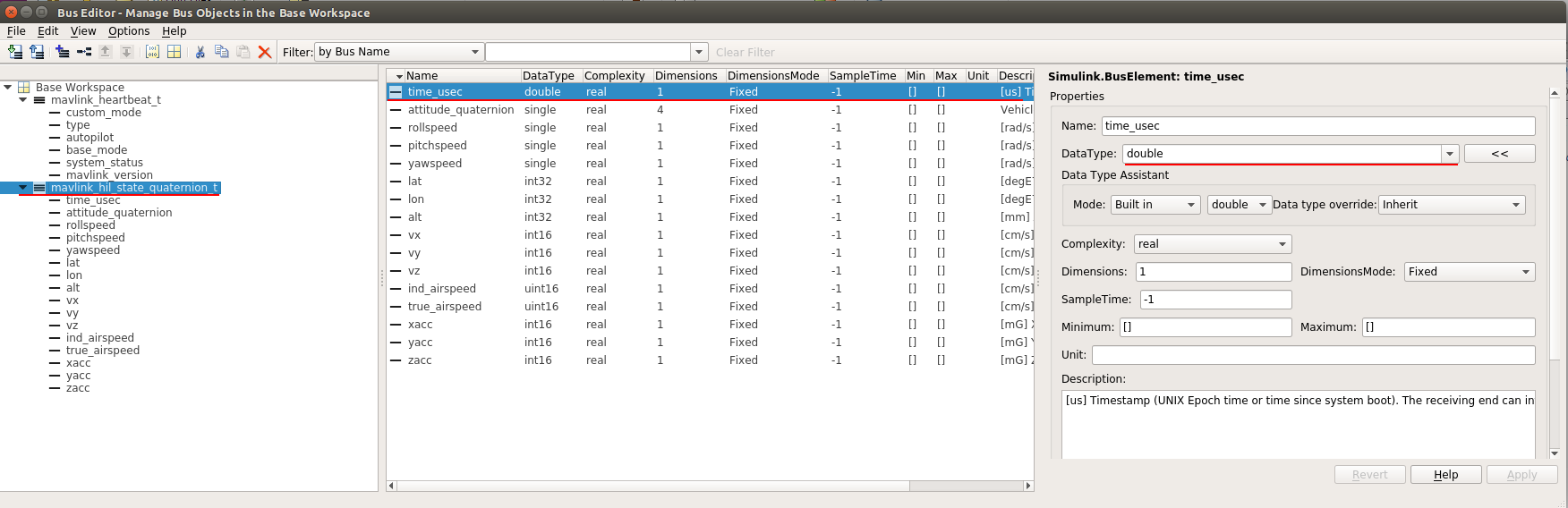


Рисунок 10. Окно управления параметрами шины

В соответствии с порядком следования полей в созданной шине необходимо соединить блоки в модели сообщения. Также размер поля можно посмотреть на сайте mavlink, где представлено декодирование сообщений (см. рис. 11). Для этого необходимо перейти по ссылке [mavlink](https://mavlink.io/en/messages/common.html) и найти интересующее сообщение.

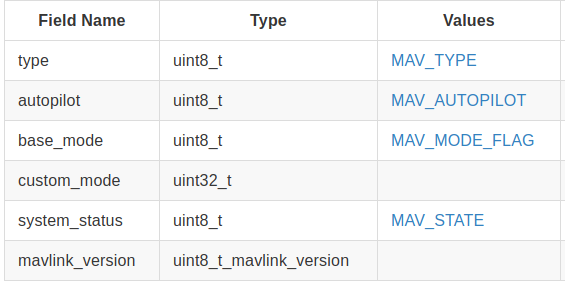


Рисунок 11. Декодирование сообщения heartbeat mavlink

Для того, чтобы добавить сгенерированную скриптом функцию (см. пункт 4.1.1) в блок s-function, необходимо в поле “S-function name” указать имя соответствующего ей файла, находящегося в папке ~/simulink/…./sfunctions (см. рис. 12).

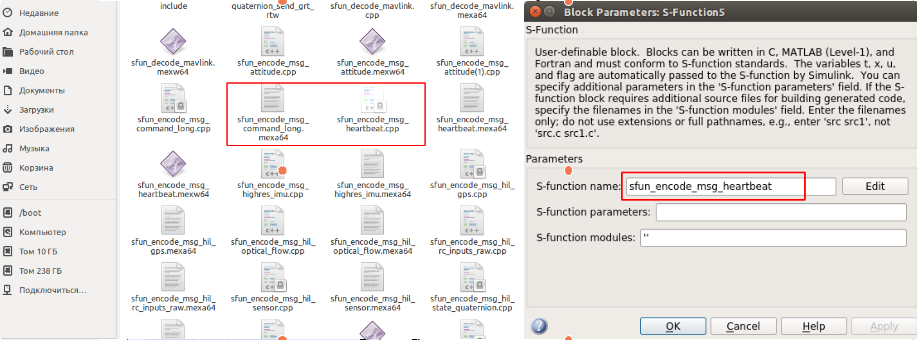


Рисунок 12. Добавление сгенерированной функции в блок s-function

Теперь необходимо сконфигурировать блок Send UDP, отвечающий за передачу и прием сообщений от автопилота. Двойным кликом по объекту вызывается окно параметров. В поле IP адрес необходимо выставить “127.0.0.1”, а в поле Remote IP Port значение 14560 (см. рисунок. 13).

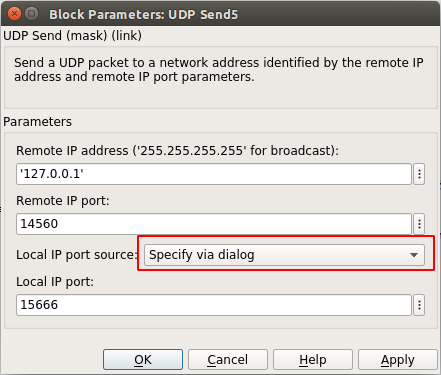


Рисунок 13. Окно параметров блока Send UDP

Также важно задать адрес собственного порта, иначе simulink по умолчанию присвоит порту случайный свободный адрес. Для каждого сообщения, которое передается по mavlink автопилоту, поле Local IP Port должно совпадать.

#### 4.1.2.2 Модель принимаемого сообщения по mavlink

Для приема сообщений по mavlink необходимо создавать модели для каждого из принимаемого сообщения.

Шаг 1. Размещение блоков в рабочей области simulink.

Создать новую рабочую область в simulink и разместить на ней компоненты, как показано на рисунке 14.

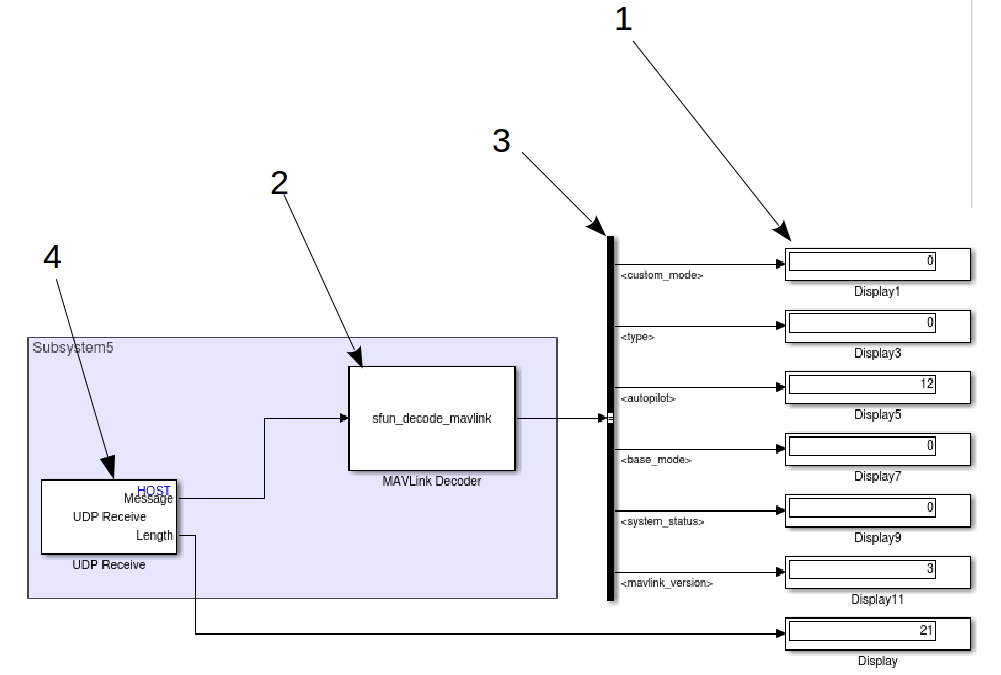
****

Рисунок 14. Модель принимаемого сообщения. 1 - блок display; 2 - блок s-function; 3 - блок bus selector; 4 - блок UDP Receive (вкладка Embedded coder)

Шаг 2. Добавление сгенерированной функции в блок s-function.

Для добавления функции декодирования в блок s-function, необходимо раскрыть меню параметров блока и в поле “S-function name” (см. рис. 15) ввести имя скомпилированной функции (см. пункт 4.2.1). Путь к ней внутри проекта выглядит следующим образом: ~/simulink/…./sfunctions/sfun\_decode\_mavlink

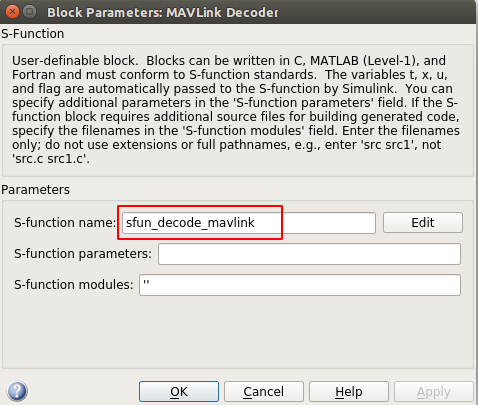


Рисунок 15. Окно параметров блока s-function

Шаг 3. Конфигурирование блока bus selector.

Настройка bus selector. Окно конфигурации компонента вызывается двойным щелчком мыши по объекту. В появившемся окне необходимо перенести все входные сигналы с шины в область selected signal. Для этого необходимо выделить все сигналы и нажать кнопку “Select>>” (см. рисунок 16), после чего кликнуть по кнопке “apply” для внесения и сохранения изменений.

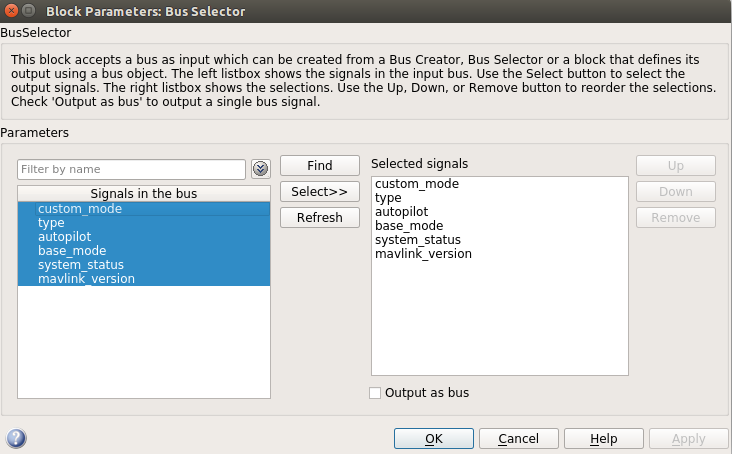


Рисунок 16. Окно конфигурации блока Selector bus

Шаг 4. Настройка блока приема UDP Receive

Для конфигурации блока UDP Receive необходимо вызвать его окно параметров, дважды кликнув по нему. В открывшемся окне (см. рис. 17), в строке “Local IP Port” нужно выставить то же значение порта, что установлено в блоке передачи UDP Send (см. пункт 3.2.2.1). В текущей реализации это значение 15666. Полю “Remote IP Address” необходимо присвоить значение '127.0.0.1'. Размер буфера передаваемого сообщения можно оставить без изменений. В поле “maximum length for message” необходимо внести значение 280. “Blocking time” выставить 0 секунд и “Sample Time" присвоить значение 0.01.

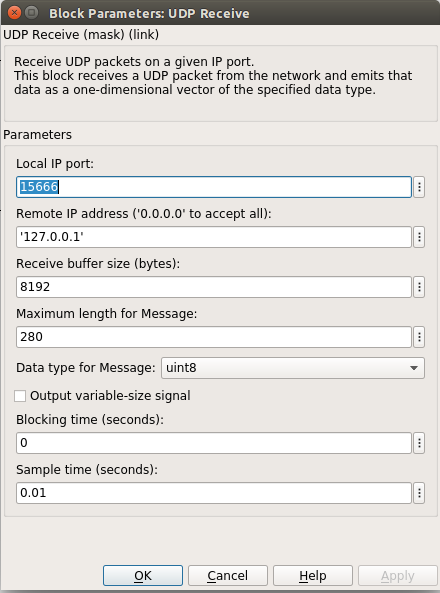


Рисунок 17. Окно конфигурации блока UDP Receive

Шаг 5.

Для отображения значения поля декодированного сообщения mavlink на рабочей области размещены блоки display. Необходимо вызвать его окно конфигурации (см. рис. 18) и выбрать для поля “Format” значение “decimal” из контекстного меню.

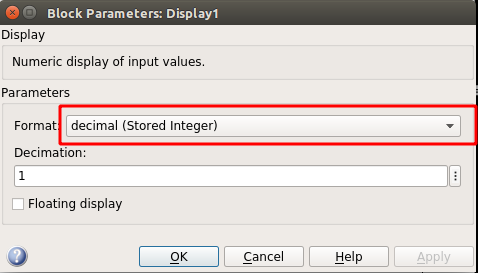


Рисунок 18. Окно конфигурации блока display

### 4.1.3 Настройка симуляции и первый запуск модели

Шаг 1. Настройка симуляции

Для запуска симуляции модели необходимо установить общее время моделирования и шаг модельного времени. Согласно алгоритму PX4 минимальное время ожидания пакета равно 5 мс, - это наименьшее время среди всех циклично передаваемых сообщений. Для выполнении вышеописанного условия в модель помещается блок simulation pace. Он позволяет управлять модельным временем и синхронизировать блоки модели между собой, что дает возможность передавать пакеты с определенной частотой. Для его конфигурации двойным кликом вызывается блок параметров (см. рис. 19).

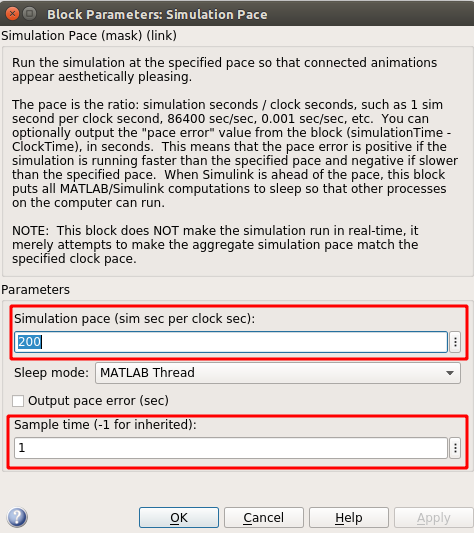


Рисунок 19. Окно конфигурирования Simulation Pace

В поле Simulation Pace задается значение 200, в поле Sample Time - значение 1.

Шаг 2. Запуск симуляции модели

Перед первым запуском модели необходимо выполнить все вышеописанные пункты. Рабочая область simulink при приеме/передачи сообщения mavlink\_msg\_heartbeat по UDP изображена на рисунке 20.

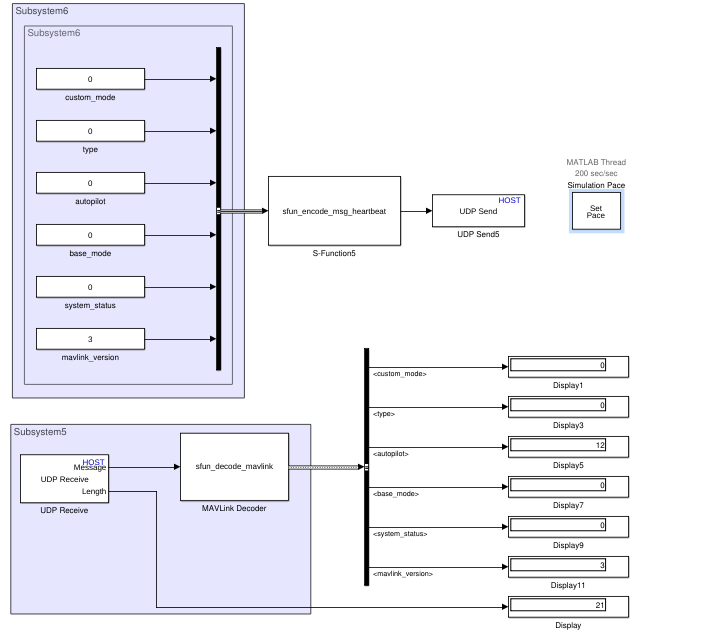


Рисунок 20. Модель приема/передачи сообщения heartbeat mavlink

Для старта симуляции модели необходимо задать общее время симуляции (см. рис. 21) и нажать кнопку “Run”, располагающуюся на панели управления simulink в верхней части окна.

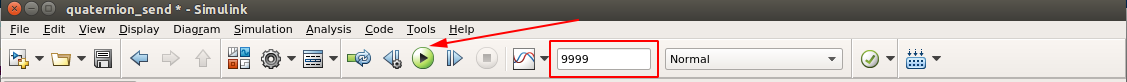
****

Рисунок 21. Панель управления simulink

При отсутствии ошибок начнется симуляция, а в нижнем углу окна simulink появится изменение значения отсчетов.

**Важно**! Необходимо сохранить модель во избежание повторения цикла построения и конфигурирования модели.

Для того, чтобы убедиться в оправке пакетов по UDP, необходимо перехватить пакеты в loopback, используя программное обеспечение для захвата и анализа сетевого трафика. В качестве такого ПО используется wireshark.

Запустите wireshark, предоставив приложению полный пакет прав. Выберите в главном окне режим “Loopback: io” (см. рис. 22).

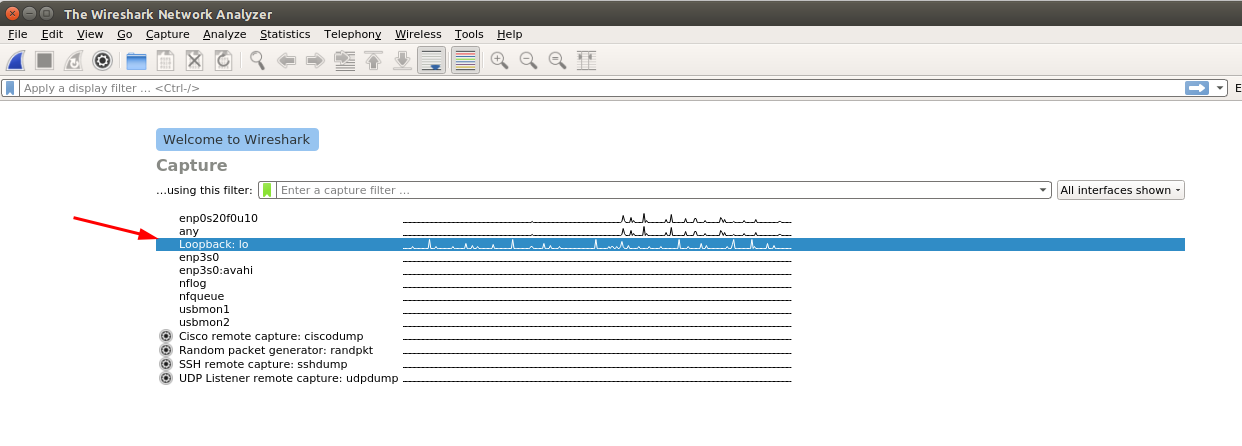


Рисунок 22. Главное окно wireshark

Если по каким-либо причинам данная строка отсутствует, необходимо проделать следующие действия :

1. в верхней части экрана в панели управления выбрать вкладку “Capture”;
2. из выпадающего контекстного меню выбрать “Option”;
3. в раскрывшемся окне выбрать строку“loopback: io” и нажать “Start” (см. рис. 23).

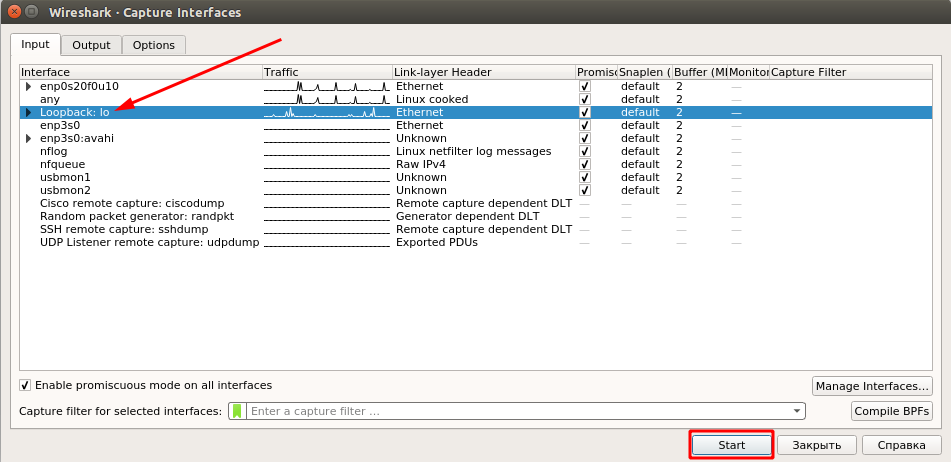


Рисунок 23. Выбор режима захвата пакетов wireshark

Если симуляция модели сообщения simulink запущена, в рабочей области wireshark будут отображаться передаваемые пакеты. Далее необходимо настроить фильтр, чтобы в рабочей области находились только пакеты, содержащие сообщение mavlink. Для этого в строку фильтра wireshark необходимо добавить (см. рис. 24) строку: udp.port== 14560.

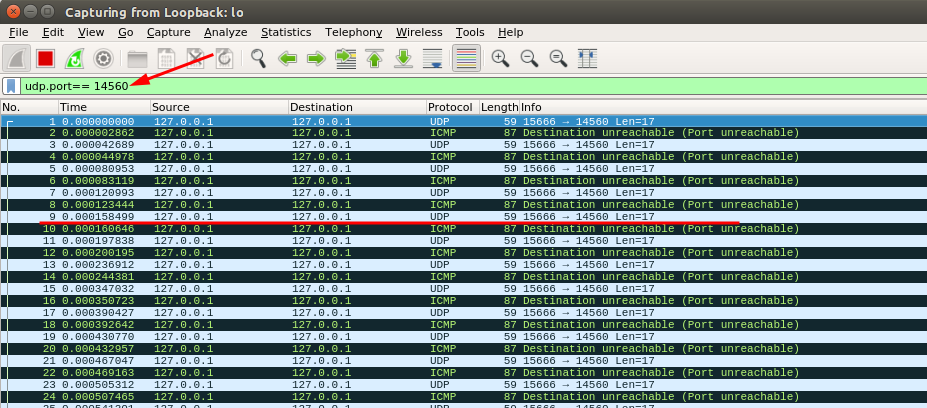


Рисунок 24. Окно рабочей области wireshark с отфильтрованными пакетами

После применения фильтра wireshark выведет в рабочую область пакеты передаваемые по UDP только от порта с адресом 15666. На рисунке 24 красной линией подчеркнут пакет, содержащий в себе сообщение heartbeat mavlink, переданный из simulink.

## 4.2 Обмен heartbeat между SITL и Simulink

Шаг 1.

Для запуска PX4 в режиме SITL необходимо перейти в папку с исходниками PX4 - ~/Firmware/, вызвать из этой папки терминал и ввести команду для сборки PX4 с целью jmavsim:

make posix\_sitl\_default jmavsim

При успешной сборке SITL будет находиться в режиме ожидания пакета с сообщением mavlink (см. рис. 25).

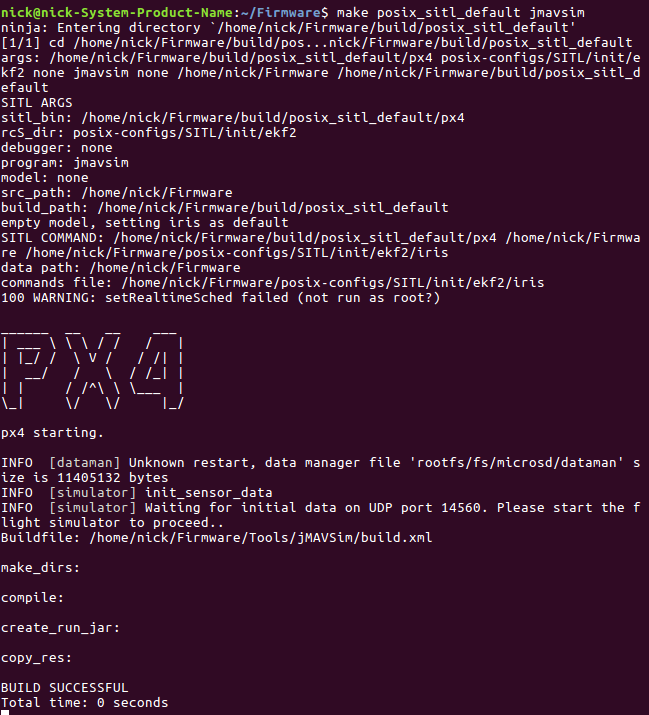


Рисунок 25. Окно терминала с успешной сборки автопилота в режиме SITL

Шаг 2.

Необходимо запустить MATLAB. Добавить в рабочую область папки и подмодули, необходимые для работы, также открыть сохраненную в пункте 3.2.1 рабочую область с шинами (см. рис. 26).

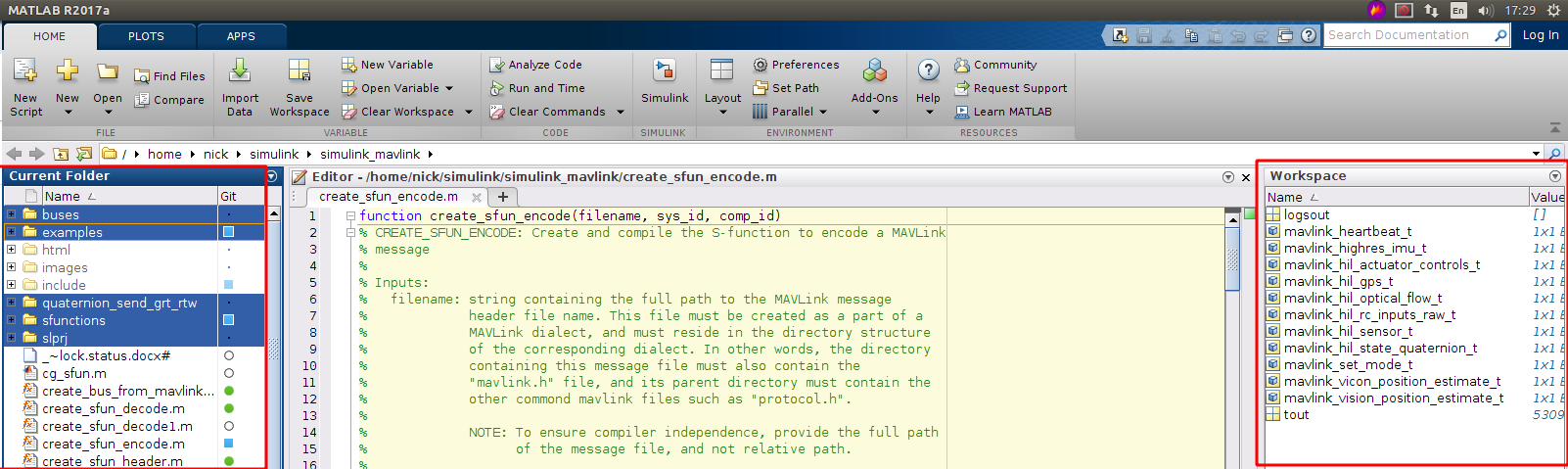


Рисунок 26. Окно MATLAB, подготовленное к симуляции модели в simulink

На панели управления MATLAB (вкладка “HOME”) необходимо активировать иконку “simulink”. В появившемся окне необходимо запустить сохраненную в пункте 3.2.3 модель (см. рис. 27).

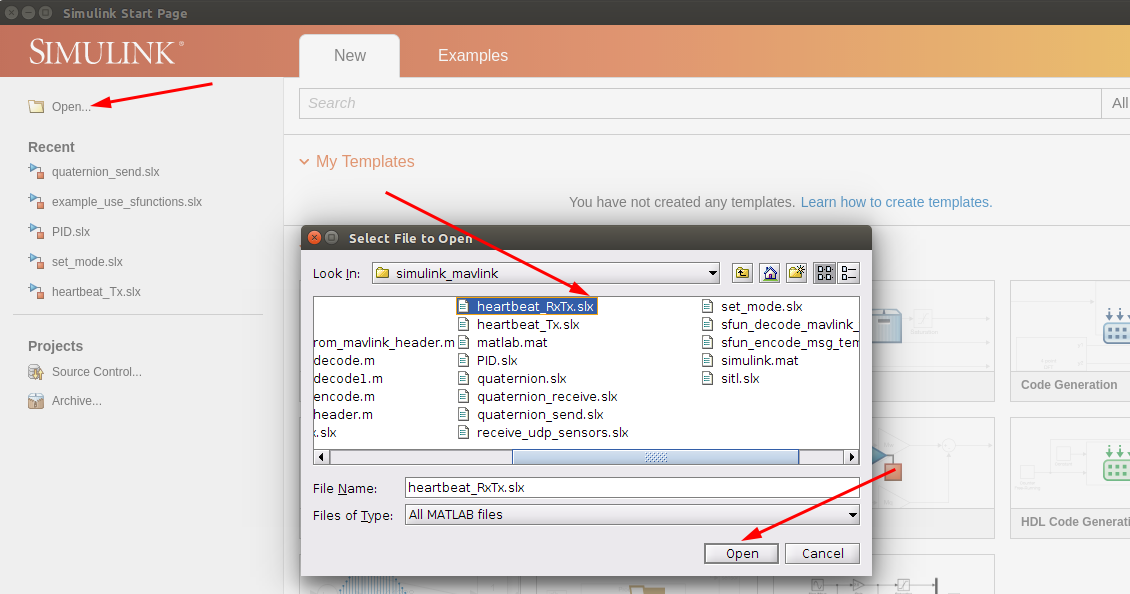


Рисунок 27. Окно simulink. Открытие сохраненной модели

В раскрывшемся окне с моделью simulink необходимо нажать на кнопку “Run”, находящуюся в панели управления, после чего начнется запуск симуляции (см. рис. 28).

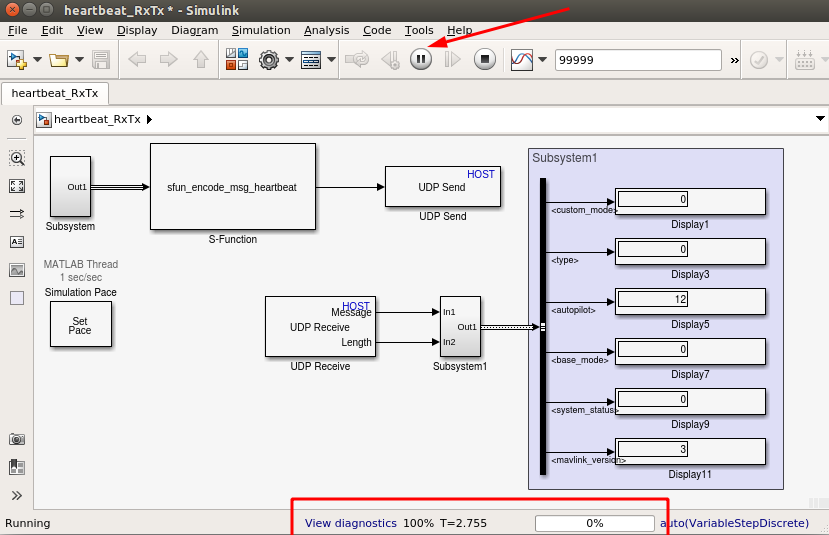


Рисунок 28. Рабочая область simulink с моделью сообщения heartbeat. Запуск симуляции

На рисунке 28 представлен пример кодирования/декодирования и приема/передачи сообщения mavlink\_msg\_heartbeat при помощи протокола mavlink, используя протокол транспортного уровня UDP.

Руководствуясь инструкцией, описанной в части 3.2.3, необходимо захватить пакеты при помощи wireshark и убедиться в обмене пакетами между системами (см. рис. 29).

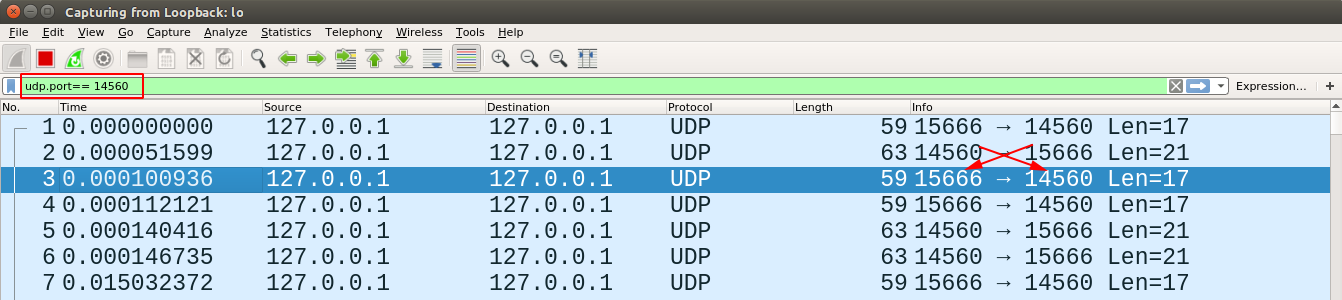


Рисунок 29. Рабочая область wireshark. Обмен heartbeat между SITL и simulink