#include <Adafruit\_SI5351.h>

Adafruit\_SI5351 clockgen = Adafruit\_SI5351();

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*

    Функция настройки Arduino (автоматически вызывается при запуске)

\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void setup(void)

{

  Serial.begin(9600);

  Serial.println("Si5351 Clockgen Test"); Serial.println("");

  /\* Инициализируется микросхема \*/

  if (clockgen.begin() != ERROR\_NONE)

  {

   /\* Возникла проблема с обнаружением микросхемы...проверьте свои подключения \*/

    Serial.print("Ой, Si5351 не обнаружен...Проверьте проводку или I2C ADDR!");

    while(1);

  }

  Serial.println("OK!");

  /\* РЕЖИМ РАБОТЫ ТОЛЬКО С ЦЕЛЫМИ ЧИСЛАМИ -> наиболее точный вывод \*/

  /\* Настройте PLLA на работу только в целочисленном режиме на частоте @ 900MHz (должно быть 600..900MHz) \*/

  /\* Установите частоту Multisynth 0 на 112,5 МГц в целочисленном режиме (разделите на 4/6/8)\*/

  /\* 25 МГц \* 36 = 900 МГц, затем 900 МГц / 8 = 112,5 МГц \*/

  Serial.println("Set PLLA to 900MHz");

  clockgen.setupPLLInt(SI5351\_PLL\_A, 36); / PLL\_A должно быть 25 МГц \* m, где m как целочисленный множитель может варьироваться от 15 до 90 /

  Serial.println("Set Output #0 to 112.5MHz");

  clockgen.setupMultisynthInt(0, SI5351\_PLL\_A, SI5351\_MULTISYNTH\_DIV\_8); / первая запятая это выход, происходит деление на 4/6/8 для достижения нужной частоты /

  /\* ДРОБНЫЙ РЕЖИМ -> Более гибкий, но с дрожанием тактовой частоты \*/

  /\* Настройте PLLB на дробный режим @616,66667 МГц (XTAL \* 24 + 2/3)\*/

  /\* Настройка Multisynth 1 на частоту 13,55311 МГц (PLLB/45,5)\*/

  clockgen.setupPLL(SI5351\_PLL\_B, 24, 2, 3); / PLL\_B должно быть 25 МГц\*(m+n/d), где m (целочисленный множитель) может варьироваться от 15 до 90, n (числитель) может находиться в диапазоне от 0 до 1 048 575, d (знаменатель) может варьироваться от 1 до 1 048 575 /

  Serial.println("Set Output #1 to 13.553115MHz");

  clockgen.setupMultisynth(1, SI5351\_PLL\_B, 45, 1, 2); / первая запятая это выход, конечная частота равна PLL / (div + n/d), div может варьироваться от 4 Для 900, n может варьироваться от 0 до 1 048 575, d может варьироваться от от 1 до 1 048 575 /

  /\* Multisynth 2 пока не используется и не будет включён, но можно \*/

  /\* использовать PLLB на частоте 616,66667 МГц, затем разделите на 900 -> 685,185 кГц \*/

  /\* затем разделите на 64, чтобы получить 10,706 кГц \*/

  /\* настроенных с помощью PLL в целочисленном или дробном режиме \*/

  Serial.println("Set Output #2 to 10.706 KHz");

  clockgen.setupMultisynth(2, SI5351\_PLL\_B, 900, 0, 1);

  clockgen.setupRdiv(2, SI5351\_R\_DIV\_64); / первая запятая это выход, дальше это допролнительный R-делитель, который помогает получить частоты <100 кГц, он ещё раз делит результат на фиксированное число 1/5/4/8/16/32/64/128 /

  /\* Включите часы \*/

  clockgen.enableOutputs(true);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*

    Функция цикла Arduino, вызываемая после завершения «настройки» (здесь должен быть ваш собственный код)\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void loop(void)

{

}

Сделать для каждой частоты отдельный блок (либо +, либо -), потом функцию, которая выбирает блок и отсылает его (куда-то)

Таблица дтмф

60150+-697 - лоу

60150+770

852

941

и

1209 - хайт

1336

1477

1633

ОБЪЯСНЕНИЕ:

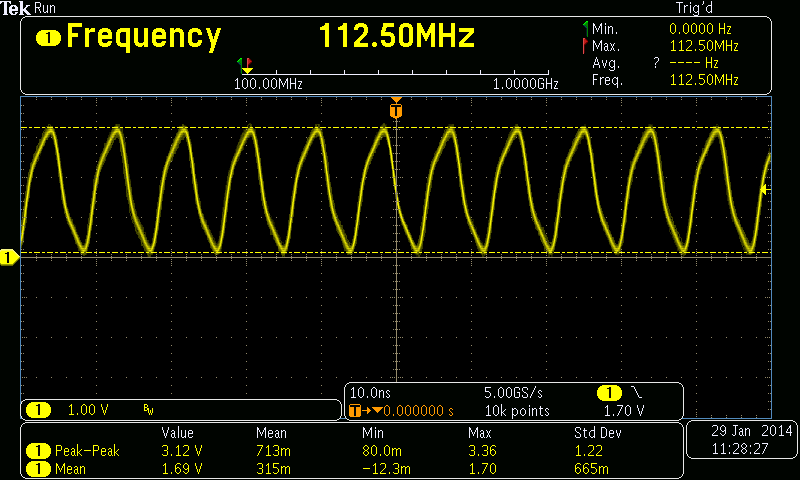
1) void - используется при объявлении функций, если функция не возвращает никакого значение при ее вызове.

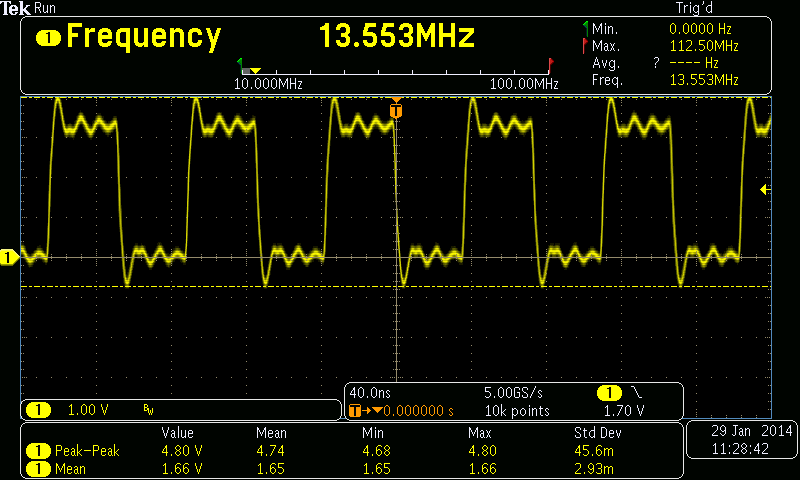
2) Serial.begin() - инициирует последовательное соединение и задает скорость передачи данных в бит/c (бод). Для обмена данными с компьютером используйте следующие значения: 300, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600 или 115200.

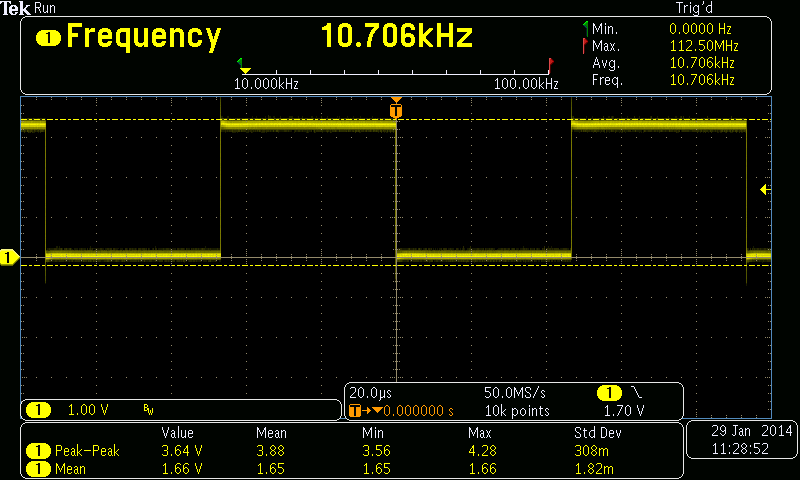
3) Serial.println(val, format) - val: данные для передачи через последовательное соединение, format: базис для целых чисел или количество знаков после запятой для вещественных.

4) Serial.print(val, format) - Передает данные через последовательный порт как ASCII текст. val: данные для передачи через последовательное соединение, format: базис для целых чисел или количество знаков после запятой для вещественных.

5) 0, 1 и 2 - это 3 независимых выходных сигнала, генерируемых тактовыми импульсами. Это прямоугольные волны от 0 до 3В.







6) Чип использует две подсистемы для генерации тактовых импульсов. Сначала он умножает тактовую частоту 25 МГц на некоторое число (настраивая ФАПЧ), затем делит эту новую частоту на другое число (настраивая делитель частоты)

<http://www.irort.ru/sites/default/files/Программируем%20Arduino.%20Профессиональная%20работа%20со%20скетчами.pdf>

<https://microkontroller.ru/arduino-projects/generator-perestraivaemoj-chastoty-na-arduino-i-module-si5351/?ysclid=m60nfuk13h982927371>

<https://eax.me/stm32-si5351/>