**NEC PROTOCOL**

It is one of Infrared protocol widely used in Remotes ,TV ,VR, Audio system etc.

* It encodes remotes keys into 32 bit frame format .
* Information is transmitted in following manner

|  |  |  |  |
| --- | --- | --- | --- |
| Address  START | Comp.Address | Command | Comp. command  E  O  M |
| LSB-MSB | LSB-MSB | LSB-MSB | LSB-MSB |
| 0-7 | 8-15 | 16-23 | 24-32 |

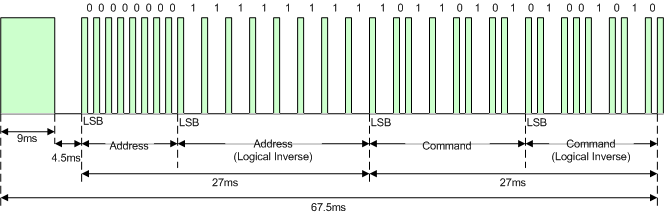
* It basically uses pulse distance encoding to transmit the signal.
* Each pulse burst is of 560 micro seconds
* Data is transmitted in 2 stages:

1. Logic 0:-

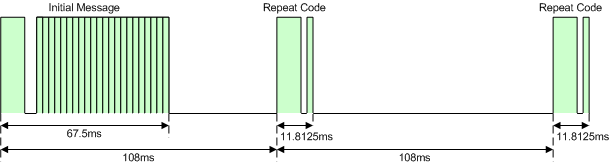
This have 560 mi s pulse burst and 560 mi s space . So the total transmit time becomes 1.125 mili seconds.

1. Logic 1:-

This have 560 mi s pulse burst and 1.6875 mi s space. So the total transmit time becomes 2.25 mili seconds.

* It have 9ms leading pulse burst(16 \* time of logical data bit).
* Followed by 4ms space.
* 8 bit->addr->8 bit->compliment->8bit ->data->8 bit->compliment
* End of message. 
* If button is kept depressed:-

1. It repeat the code.
2. It will continue to send out code after every 108ms interval of time until key is released.

* 
* 9ms leading pulse burst
* a 2.25ms space
* a 562.5µs pulse burst to mark the end of the space.

**CUBEMX INITIALIZATION**

* To initialize NEC in stm32 microcontrollers we will use GPIO and Timers.
* Select any GPIO pin in External Interrupt mode with rising/falling edge trigger detection.
* We are using interrupt mode in GPIO to read data from signal.
* Enable timer 1 with clock source as internal clock.
* Disable all channel before setting up the timer.
* SET:-

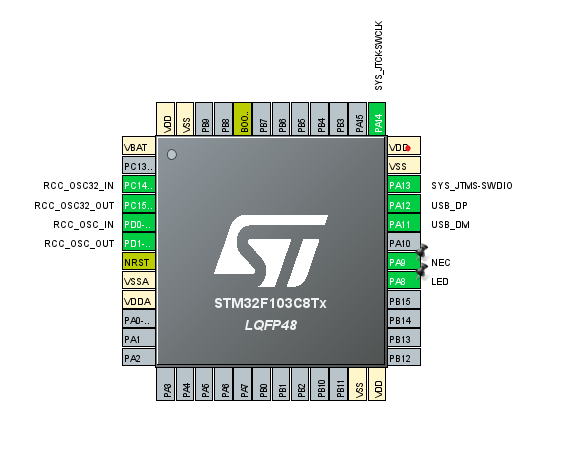
Prescaler :-719

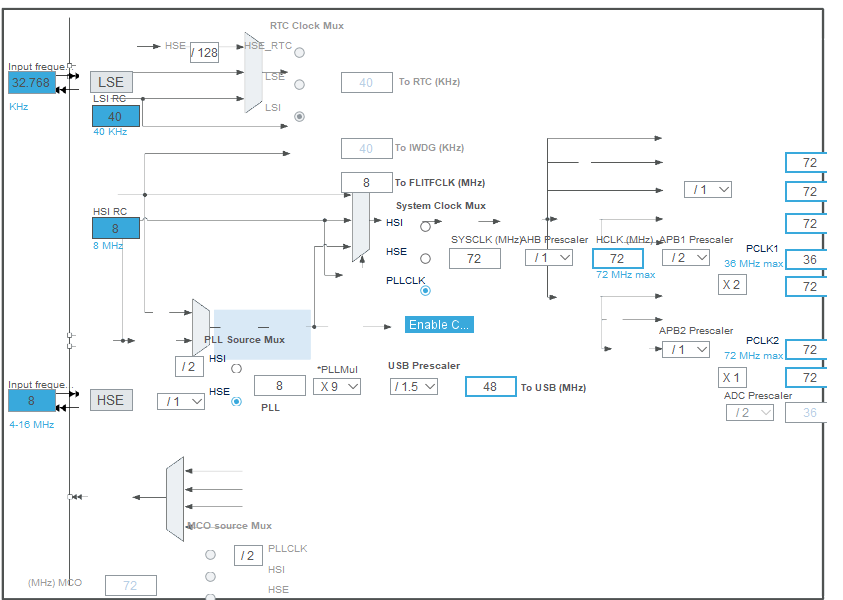
Count mode:- up

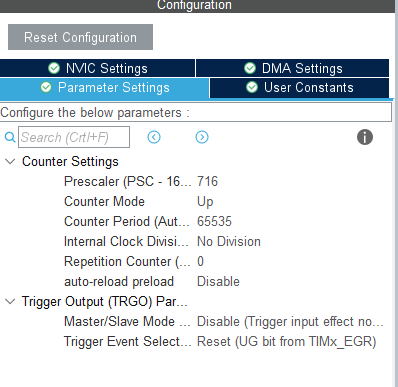
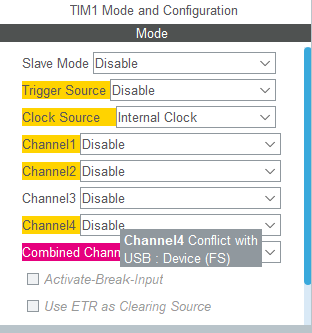
Counter period :-65535

Disable master /slave mode.

* After initializing following above steps we can easily initialize NEC in stm32.







**NEC Function:**

We have created the structure of NEC protocol consisting many structure member such as:

**struct** NEC {

uint8\_t addr;

uint8\_t addr\_inv;

uint8\_t cmd;

uint8\_t cmd\_inv;

uint8\_t i;

uint8\_t init\_seq;

uint8\_t gpio;

uint16\_t count;

uint8\_t repeat;

uint8\_t complet;

};

**struct** NEC NEC1;

NEC1 is object of NEC structure.

**GPIO EXTI CALL BACK function:-**

I have a digital input GPIO line where I need an interrupt whenever its input changes. In STM32CubeMX I set this pin to an EXTI line and set the interrupt to trigger on both rising and falling edges.

**The prototype of the callback is:**

**void HAL\_GPIO\_EXTI\_Callback(uint16\_t GPIO\_Pin);**

In this function we are checking all the GPIO pinsand if GPIO pin== NEC pin then we are reading the Input at NEC pin using

**HAL\_GPIO\_ReadPin(NEC\_GPIO\_Port, NEC\_Pin);**

And store the input into NEC1.gpio

Then we have applied various condition to check our data/signal and update our structure variable until we receive correct signal.

**Timers(GPT):**

The General Purpose Timer (GPT) has a 32-bit up-counter. The timer counter value can be captured in a register using an event on an external pin. The capture trigger can be programmed to be a rising or/and falling edge. The GPT can also generate an event on the DO\_CMPOUTn pins and an interrupt when the timer reaches a programmed value. The GPT has a 12-bit prescaler, which provides a programmable clock frequency derived from multiple clock sources.

**Timers callback function:-**

In this function when timer instance is TIM1 it will set LED Pin and reset all the structure variable.

**OUTPUT:-**

After completing the code we were able to receive all the signal of IR Remote using stm32 via NEC protocol.

**Testing:-**

1. We can use profession tools.
2. While debugging we can check whether Registers value is updating when remote’s depressed.