# Computer Systems 414

13 April 2021

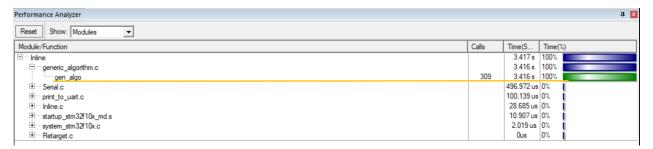
Walt Deyzel: 21750793

### Code:

### Busy wait method.

Module/Function	Calls	Time(Sec)	Time(%)
⊟ Inline		4.988 s	100%
⊟ Serial.c		4.427 s	89%
SER_Init	2	456.833 us	0%
\$Sub\$\$_rt_entry	1	0.083 us	0%
SER_PutChar	76438	4.426 s	89%
SER_waitLess	0	0us	0%
SER_GetChar	0	0us	0%
generic_algorithm.c		553.580 ms	11%
⊕ print_to_uart.c		7.795 ms	0%
startup_stm32f10x_md.s		10.907 us	0%
⊕ Inline.c		4.769 us	0%
system_stm32f10x.c		2.019 us	0%
• Retarget.c		0us	0%

### Flag control print method



## **Busy Wait**

# secret raids on Al-Quaeda, and I have over NO continued Mills. I as trained in portilla warfare and I'w the top a person. The storms that wipse out the pathwise little thing you callyour life. You've dead stready, kid, I can be any person of the continued of th

Flag control



The *busy wait* method writes the entire message via **UART** before executing the generic algorithm method that simulates the CPU tracking of the IP address. This uses 89% of the program execution time according to the performance analyser to send the message and 11% to track the IP address.

The *Flag control* method sends *one character* at a time (keeping track of index via static int) from the message then executes the generic algorithm method that simulates the CPU tracking of the IP address. This method uses basically 0% of the execution time to send the message which leaves approximately 100% of the execution performance to the CPU. Making it a more efficient method to track the IP address but takes longer to send the message to the person on the other end of the connection.

A better method would be to run both the generic algorithm and sending message function in parallel. So that neither function must wait for the other to finish.