Prelab Answer Sheet: Contiki-NG and Cooja

Answer 1: 8/10

Answer 2:
Stack: Special part of Application memory which store temporary variable created by function. Every function has its own stack area. When the task of the function is complete variable will be automatically erased.

Heap: Part of memory that is not tightly managed by CPU. To allocate memory on the Heap, you must use malloc() or calloc(), which are built in C. You have to free up the allocated memory by yourself.

Stack overflow: Stack overflow will generate a “Segmentation fault” . Normally if we declare large number of variable or matrix or infinitely call a recursive function.

Answer 3:
‘static’ variables are initialized only once. ‘static’ function does not need object to call.

Global static variable and static function has no impact another file. Global static variable and static variable inside function works same fashion except static variable inside function not accessible from outside that function.

Answer 4:

Divide by 16

```
void divideBy16_BinaryOperator(){
   int16_t K = 112;
   K = K >> 4;
   printf("result => %u",K);
}
```

K%16

```
int16_t modBy16(int16_t number){
        number = (number &  15);  //n % 2^i = n & (2^i - 1)
        return number;
}
```

Prelab Answer Sheet: Contiki-NG and Cooja

Answer 5:

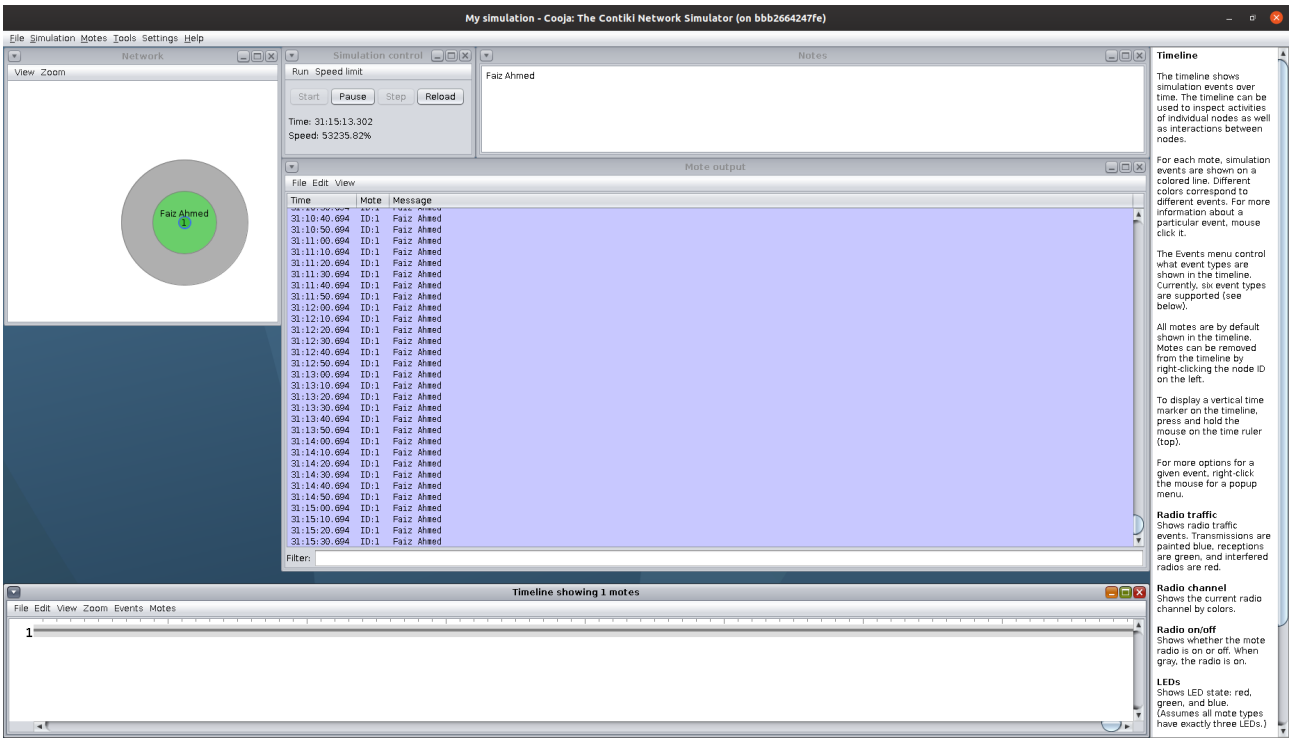

Figure 1: Printing full name

Answer 6:

etimer: Event timer is used to schedule an event to Contiki processor after a period of time while the rest of the system can work or enter low power mode.
- essential to conserve power consumption at the sensor node.

rtimer: provides a scheduling of real time system. Rtimer pre-empt any running process in order to let the real time tasks execute at the scheduled time.
- used in X-MAC implementation where the radio needs to be turn on or off at the schedule time without delay.

Prelab Answer Sheet: Contiki-NG and Cooja

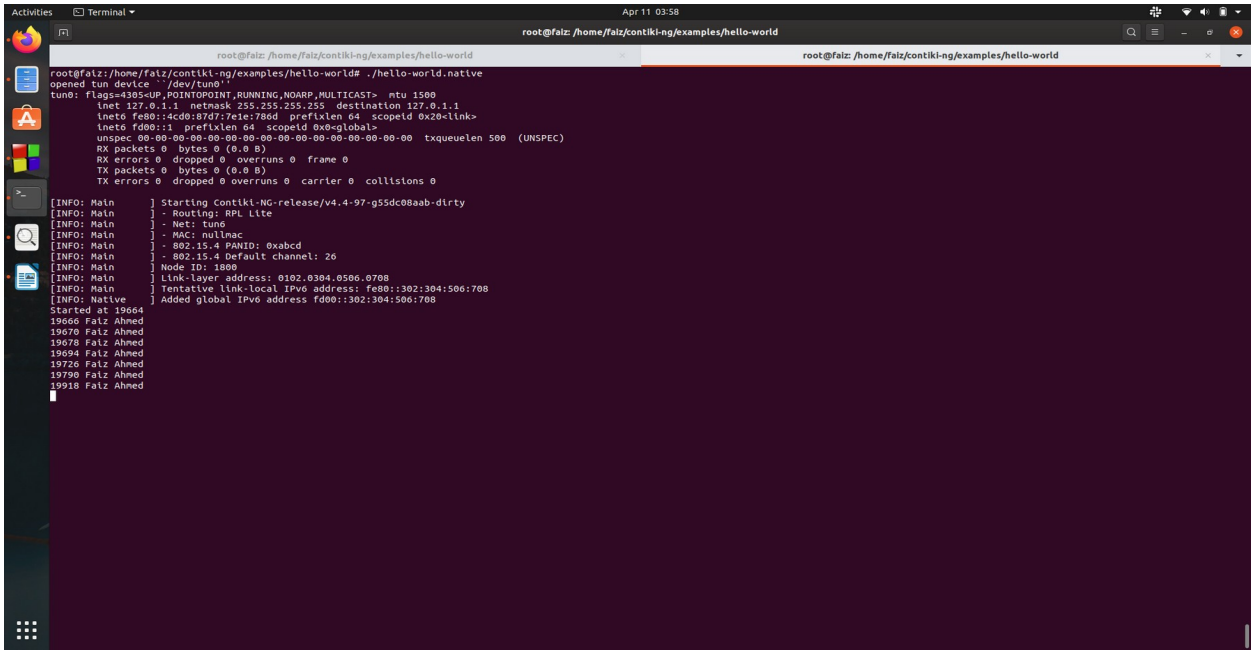

Figure 2: Printing current time in Second and Name in console

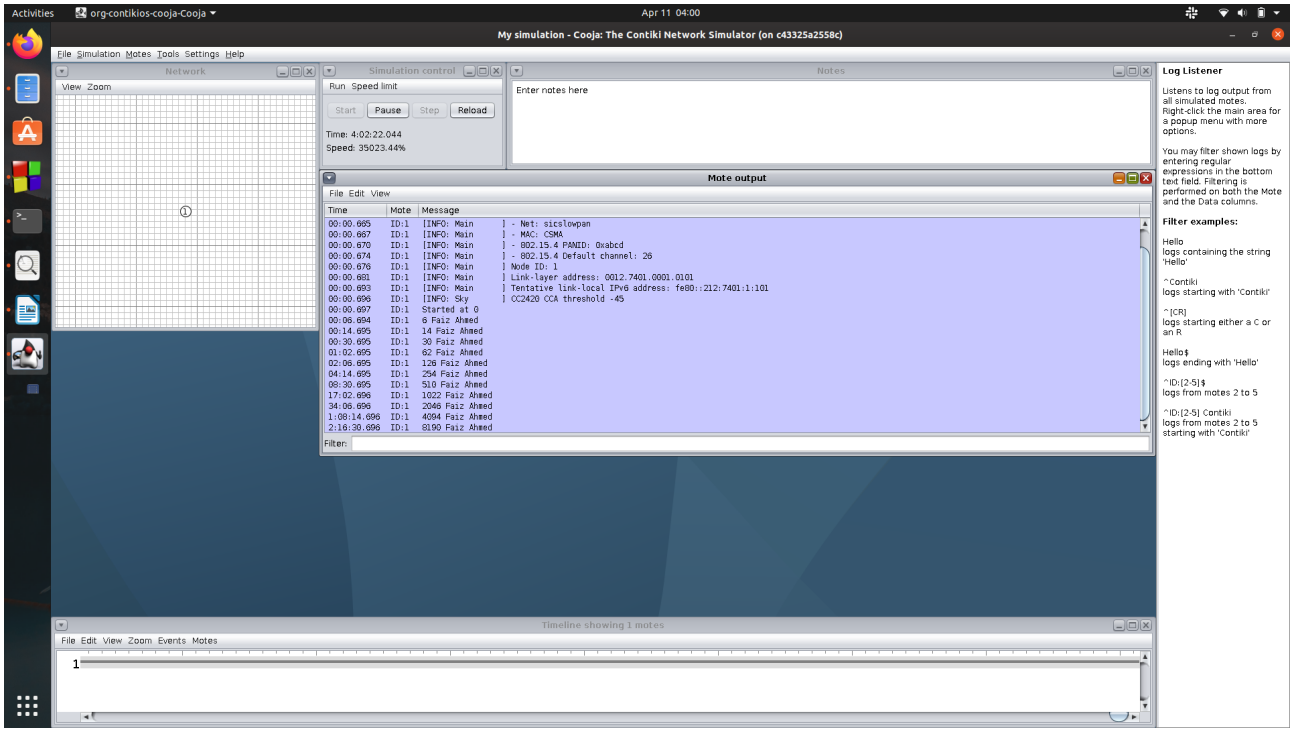

Figure 3: Printing time in seconds and Name in Simulator

Prelab Answer Sheet: Contiki-NG and Cooja

Answer 8:
Receive packet: For both Uincast and Broadcast Nullnet will call a user defined function of type '`nullnet_input_callback`'. The function arguments provide a pointer to the payload, the payload length, and the link-layer addresses of the source and destination.

Example :

```
#include "net/nullnet/nullnet.h"
//
void input_callback(const void *data, uint16_t len,
  const linkaddr_t *src, const linkaddr_t *dest)
{
// perform your operation here...
}
//...
/* At process initialization */
nullnet_set_input_callback(input_callback);
```

Send Packet: To send a packet with NullNet we have to first set 'nullnet_buffer' to your buffer and set 'nullnet_len'. Finally use NETSTACK_NETWORK.output(NULL) for broadcast or NETSTACK_NETWORK.outpur(&dest_address) for unicast.

Answer 9:
Unicast : This protocol is used to send data one point to another point. One sender and one receiver.

Broadcast : One sender multiple receiver. In this case there is just one sender, but the information is sent to all connected receivers.7

Optional Answer
I will follow the following steps.

Broadcasting 'count' data in every 10 second interval

1. Make 'count' variable as global
2. change the type of 'count' with '`int16_t`' a
3. initialized nullnet_buf = (`int16`_t *)&count
4. redefine 'SEND_INTERVAL' as (10 * CLOCK_SECOND)

Receive Data

1. inside 'input_callback' method, rename the 'count' variable like '`int16_t counter`'
2. Below printing all logs do the following
   if(counter > count){
   count = counter;
   }

Problems:
- Firstly, global variable 'count' is not ***thread safe***, concurrency related problems may occur.

Prelab Answer Sheet: Contiki-NG and Cooja

- Secondly, ‘count’ could be inconsistent in every “Tmote Sky motes” , because two packet will not be processed at the same time and while processing one packet another packet may drop, so there is a possibility of inconsistency.

Submitted by
Faiz Ahmed (stu225473)
Mutasim Fuad Ansari(stu225365)