Introduction to IoT

School Year 2023-2024

Valsalice

Course Structure

1	Introduction and Basics	
2	Basic Data Types and Operators	
3	Control Structures Pt. 1	
4	Control Structures Pt. 2	
5	Functions and Scope	
6	Arrays	
10	Preprocessor and Macros	DEC 19
11	Custom Data Types	JAN 9
7	Introduction to Contiki-NG and nRF52840	NOV 28
8	Sensing and Actuating with Contiki-NG	DEC 5
9	Basic Communication and Networking	DEC 12
12	Introduction to RPL and Network Routing	JAN 16
13	Challenges in Wireless Communication	JAN 23
14	Advanced Protocols: TSCH and 6TiSCH	JAN 30
15	Advanced Topics in Wireless Communication	FEB 6
16	Reliable Data Transfer Challenge	FEB 20, FEB 27, MAR 5

= Core Topics = Optional Topics

Open your Virtual Machines

1. Turn on your Laptops
2. Login to Windows using “User”
3. Open the **Virtual Box** program
4. Add a new Virtual Machine (**Ctrl + A**)
5. Open the **VirtualBox** folder ⚠ (**NOT** the .VirtualBox)
6. Select the **nRF52840LAB** file
7. Click **Start**

Prepare the Coding Environment

1. Start the Virtual Machine **nRF52840LAB**
2. Log-in using credentials:

   Username: **ubuntu**

   Password: **ubuntu**

3. Open **Visual Studio Code** (use the App bar on the left)

Prepare the Coding Environment

4 From the Terminal:

```
make setup
```

5

```
○ → valsalice-iot-23 git:(master) make setup
  Enter your username:
```

6

```
Password
```

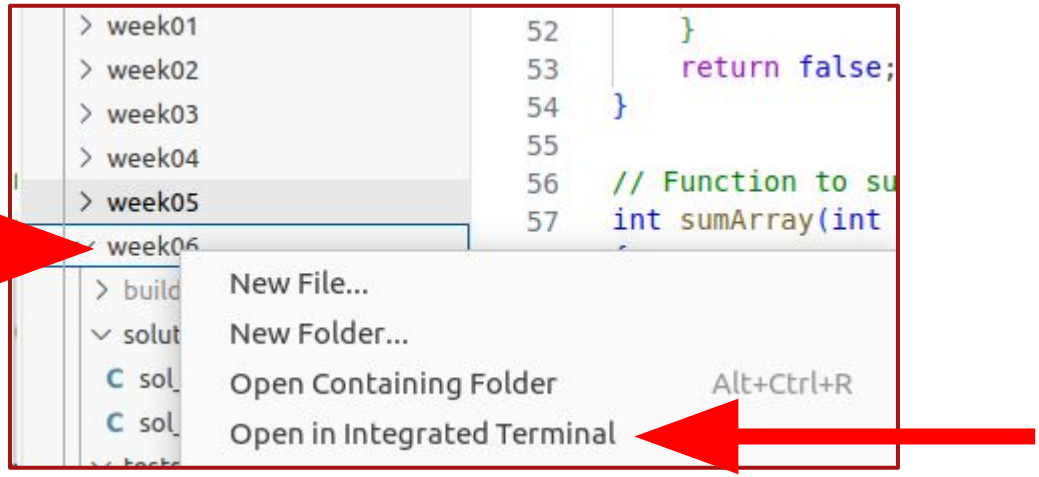
7

```
✅ Repository setup complete!
```

⚠ If you see **any (yellow) errors** input the credentials again

Prepare the Coding Environment

7 Open the **week07** folder in the terminal



8 You should see the following in the terminal:

```
week06 git:(master)
```

Recap: Data Types

C has a number of primitive data types:

int `42` `1200` `1_200` `-3`

float `3.14` `0.00001` `-2.1`

char `'A'` `'@'` `'\n'`

bool `true` `false`

Strings are *NOT* a primitive data type, and have special syntax.

strings `"Hello"` `"A"` `"I am a full sentence!"`

Recap: Variables

A variable is a named container that stores data or values.

```
int x = 42;
float y = -0.12;
char w = 'A';
char z[50] = "Full sentence";
```

Booleans require a custom include statement:

```
#include <stdbool.h>
bool hello = true;
```

Recap: Boolean Operators

Greater than	>
Greater or equal than	>=
Less than	<
Less or equal than	<=
Equals	==
Not equals	!=
Not	!

Recap: Chaining Comparisons

- **and** (both must be true)

```
true && false
```

```
(5 < 6) && (5 < 10)
```

- **or** (either must be true)

```
true || false
```

```
(5 < 3) || (5 < 10)
```

- **not** (negation)

```
!true
```

```
!(5 < 3)
```

Recap: If-Statement chaining

You can chain multiple conditions with **`else if`**.

What is the difference between these two snippets of code?

```c
int num;
scanf("%d", &num);

if (num < 3) {
    printf("Small number\n");
} else if (num < 10) {
    printf("Medium number\n");
}
```

```c
int num;
scanf("%d", &num);

if (num < 3) {
    printf("Small number\n");
}
if (num < 10) {
    printf("Medium number\n");
}
```

Recap: While-Loops

Repeat parts of your code!

```
int num;
printf("Input a number greater than 100: ");
scanf("%d", &num);

while (num <= 100) {
    printf("Wrong number, try again: ");
    scanf("%d", &num);
}

printf("Well done!\n");
```

Recap: For-Loops

Repeat a **specific** amount of times!

```
int x;

for (x = 1; x <= 5; x++) {
    printf("Hello %d\n", x);
}
```

```
int x = 0;

while (x < 5) {
    x += 1;
    printf("Hello %d\n", x);
}
```

Recap: Arrays

Modifiable containers for data.

With **variables**:

```
int num1 = 42;
int num2 = 100;
int num3 = 10;

printf("%d\n", num1);
printf("%d\n", num2);
printf("%d\n", num3);
```

With a **list**:

```
int array[] = {42, 100,
10};

for(int i = 0; i < 3; i++)
{
    printf("%d\n",
array[i]);
}
```

Recap: Arrays

Anatomy of an array:

1. Uses square brackets in the type declaration **[ ]**
2. Uses curly brackets for initialization **{ }**
3. Elements separated by comma **,**

```
int array[] = {42, 100, 10};
```

```
float array2[] = {0.23, 4.1};
```

```
char array3[] = {'a', 'Z'};
```

Recap: Accessing Array Elements

To access array elements you can use the **[index]** operator.

NOTE: List indices start from **0**

index:	0	1	2	3	4
`int array[] =`	`{17,`	`28,`	`33,`	`56,`	`6};`

```
printf("%d\n", array[0]);
```

```
printf("%d\n", array[3]);
```

Assigning Array Elements

To assign array elements you can use the **[index]** operator on the left-hand-side of a statement (like a variable)

```
int array[] = {17, 28, 33, 56, 6};
array[3] = 100;
array[2] = -7;
```

```
printf("%d\n", array[0]);
```

```
printf("%d\n", array[3]);
```

Simple Recap Exercises

Fill in all functions in (**`simple.c`**):

- `int find_max(int a, int b)`
  Find the maximum of two numbers
- `int calculate_sum(int n)`
  Calculate the sum of integers from 1 to n
- `int array_sum(int arr[], int size)`
  Find the sum of elements in an integer array

To execute: `make simple.run` **To test:** `make simple.test`

Save remotely your Changes

1. `make save`

2. Password

   Git: https://aspina@git.spina.me (Press 'Enter' to confirm or 'Escape' to cancel)

3. ✅ Changes committed and pushed. All done!

Advanced Recap Exercises

Fill in all functions in (**`advanced.c`**):

- `void replace_even_with_8(int arr[], int size)`
  Replace integers at even indices with 8
- `int find_max_element(int arr[], int size)`
  Find the maximum element in an integer array
- `void reverse_array(int arr[], int size)`
  Reverse an integer array

To execute: `make advanced.run` **To test:** `make advanced.test`

Save remotely your Changes

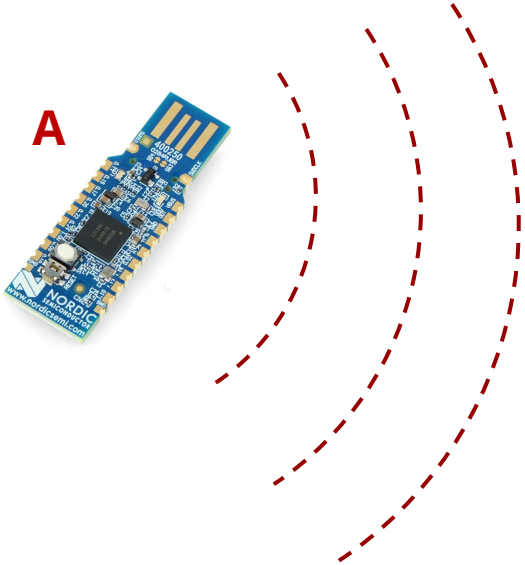
1. `make save`

2. Password

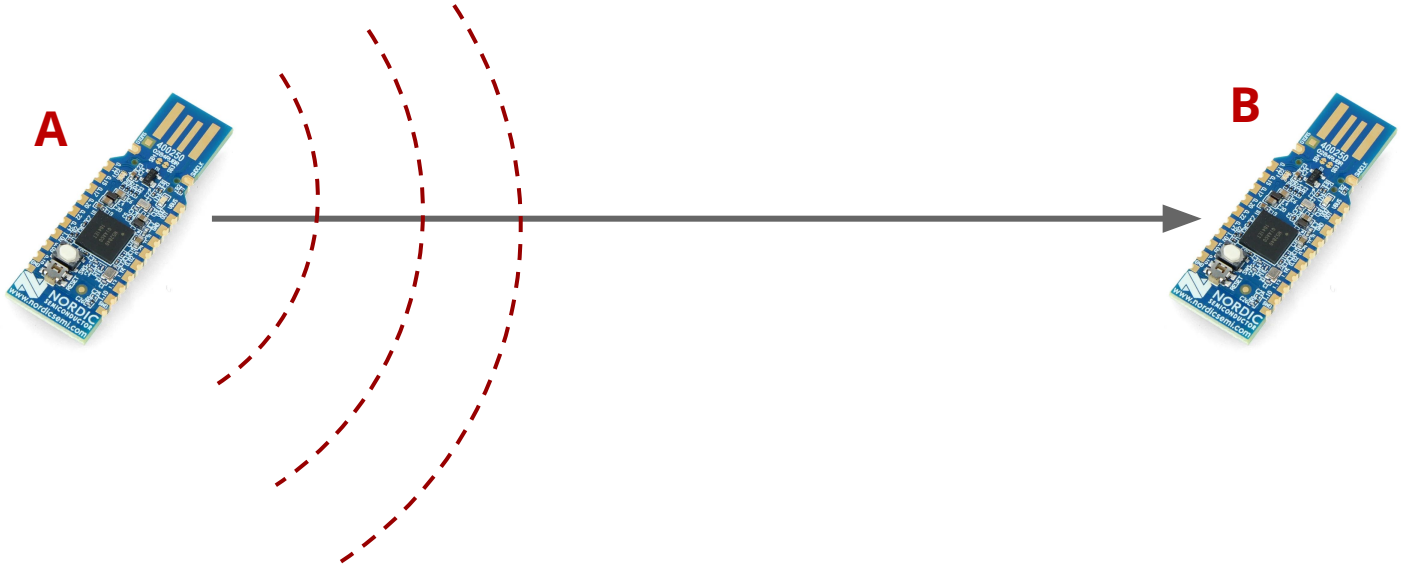
   Git: https://aspina@git.spina.me (Press 'Enter' to confirm or 'Escape' to cancel)

3. ✅ Changes committed and pushed. All done!

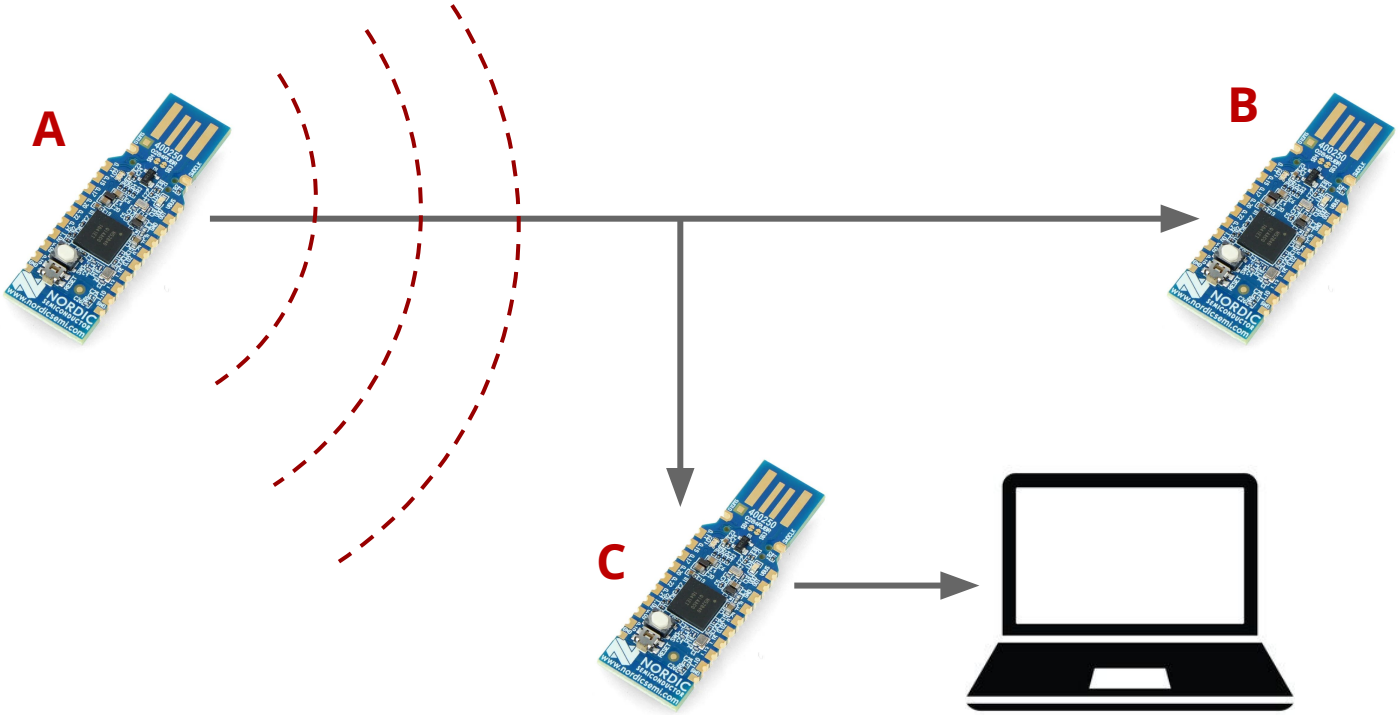
Live Demo



Live Demo



Live Demo



What is IoT?

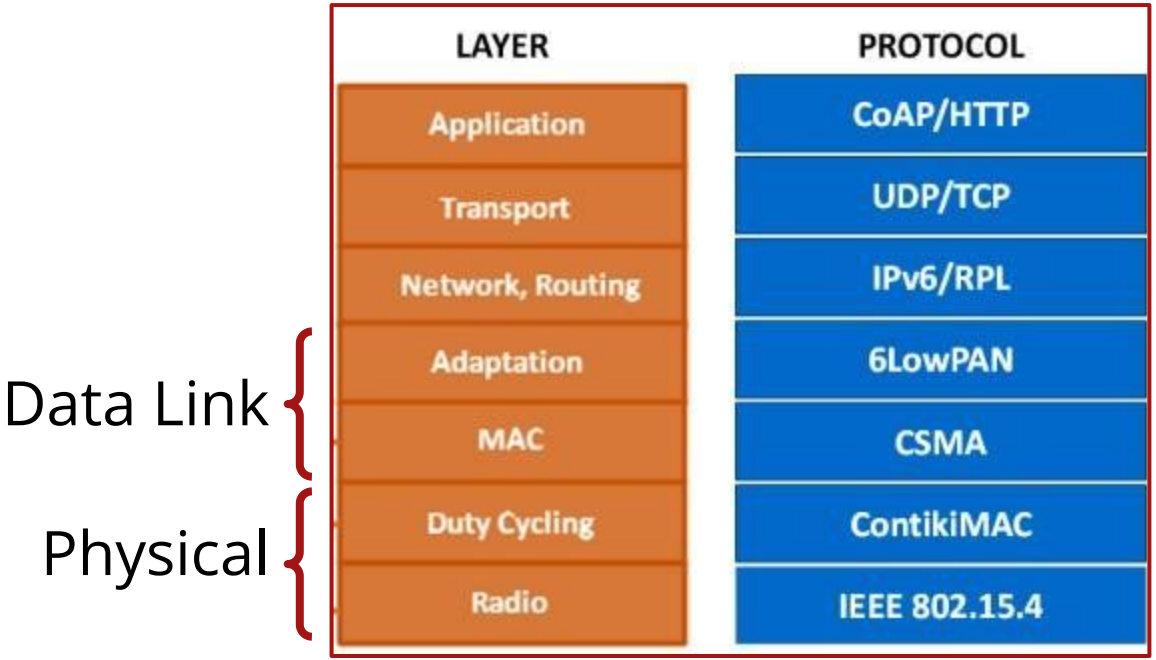
IoT stands for "**Internet of Things**." It refers to a **network of interconnected physical devices**, vehicles, buildings, and other objects embedded with **sensors**, software, and connectivity, allowing them to **collect and exchange data** over the internet.

What are Wireless Sensor Networks?

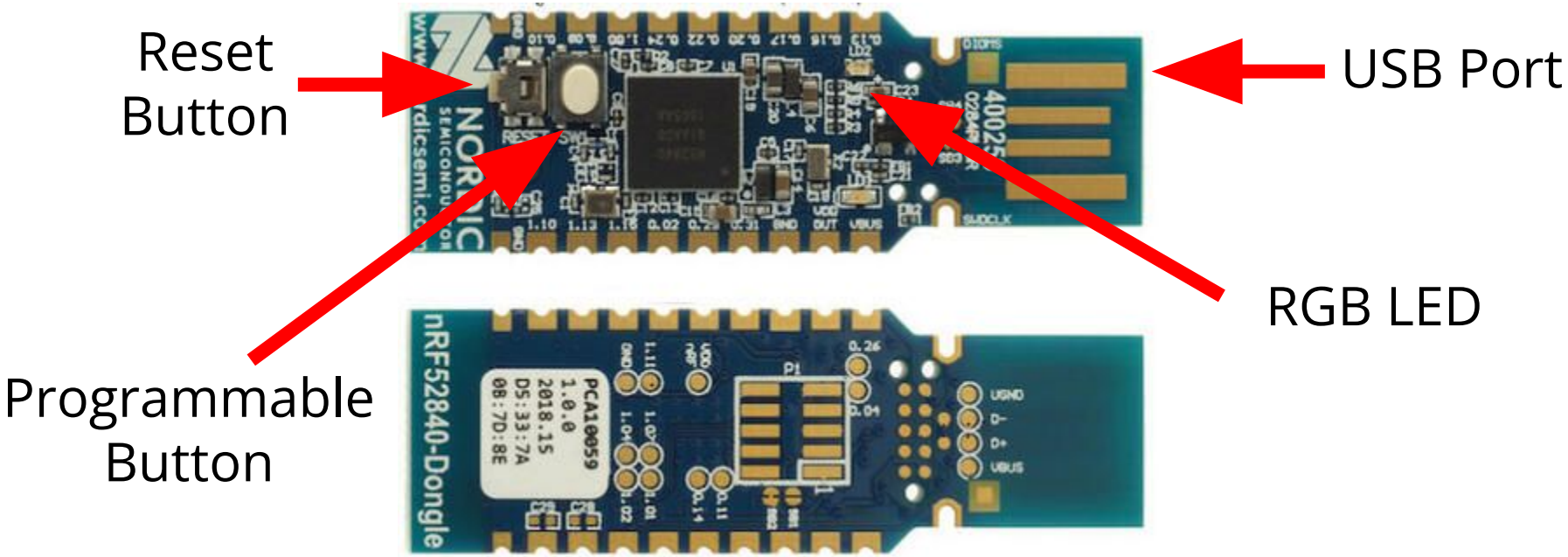
It refers to a **network of interconnected physical devices**, embedded with **sensors**, software, and connectivity, allowing them to **collect and exchange data** over the radio.

What is Contiki?

Contiki is an OS with configurable network layers:



What is the nRF52840?



Turning the LED On

1. Attach the **nRF52840** chip to your laptops

⚠ Ensure the device is in **bootloader mode** (blinking red light)

Reset Button

3. Program the firmware

```
make blinker.dfu-upload
```

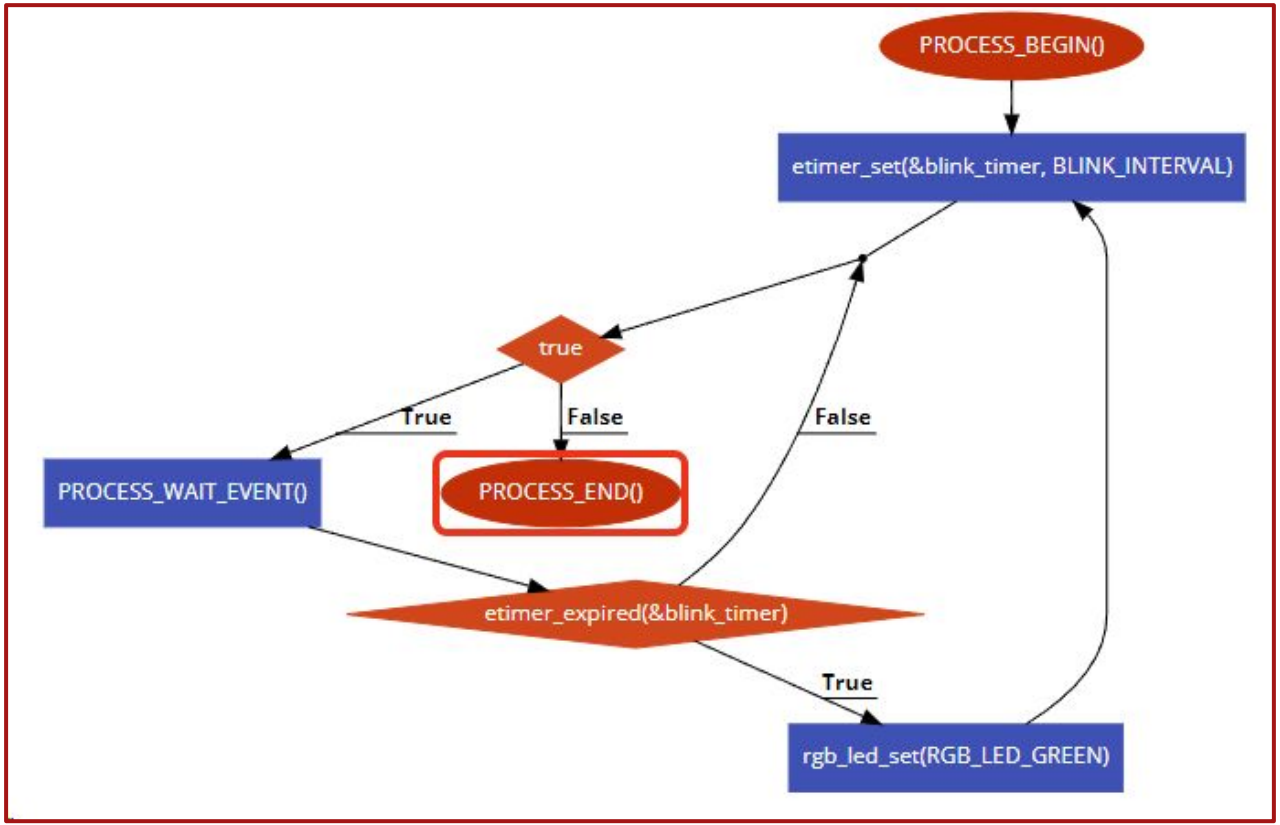
4. Attach to the serial output

```
make login
```

Anatomy of a Contiki-NG Program

```
1  PROCESS_THREAD(simple_led_program, ev, data) {
2      static struct etimer blink_timer;
3      PROCESS_BEGIN();
4      etimer_set(&blink_timer, BLINK_INTERVAL);
5      while (true) {
6          PROCESS_WAIT_EVENT();
7          if (etimer_expired(&blink_timer)) {
8              rgb_led_set(RGB_LED_GREEN);
9              etimer_set(&blink_timer, BLINK_INTERVAL);
           }
       }
10     PROCESS_END();
   }
```

Anatomy of a Contiki-NG Program



Make the LED blink

1. Attach the **nRF52840** chip to your laptops

⚠ Ensure the device is in **bootloader mode** (blinking red light)

Reset Button

3. Program the firmware

```
make blinker.dfu-upload
```

4. Attach to the serial output

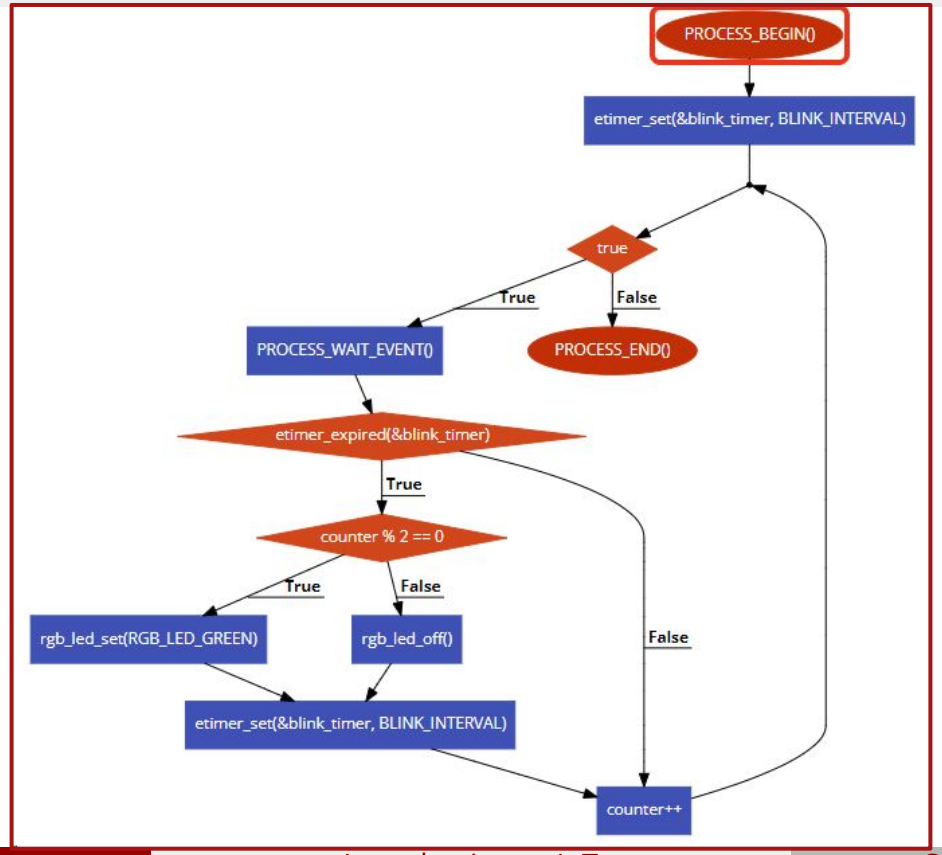
```
make login
```

Blinking Light Exercise

```
PROCESS_THREAD(simple_led_program, ev, data) {
    static struct etimer blink_timer;
    static int counter = 0;
    PROCESS_BEGIN();
    etimer_set(&blink_timer, BLINK_INTERVAL);
    while (true) {
        PROCESS_WAIT_EVENT();
        if (etimer_expired(&blink_timer)) {
            if (counter % 2 == 0) {
                rgb_led_set(RGB_LED_GREEN);
            } else {
                rgb_led_off();
            }
            etimer_set(&blink_timer, BLINK_INTERVAL);
        }
        counter++;
    }
    PROCESS_END();}
```

1 2 3 4

Blinking Light Exercise



Save remotely your Changes

1. `make save`

2. Password

   Git: https://aspina@git.spina.me (Press 'Enter' to confirm or 'Escape' to cancel)

3. ✅ Changes committed and pushed. All done!

End of Class

See you all next week!