Mikrobitowcy

Autorzy: Dawid Osuchowski, Filip Kłębczyk

Lekcja 11:
Micro:bit - radio

Podczas tej lekcji uczniowie zapoznają się z modułem radio Micro:bita. Dowiedzą się, jak wysyłać i odbierać dane oraz jakie konstrukcje stosować przy tych operacjach. Zdobytą wiedzę wykorzystają do stworzenia programu - czatu obrazkowego pomiędzy dwoma płytkami.

Cele lekcji:
Uczeń powinien:
- Znać pojęcia: radio, kanał radiowy, konfiguracja, transmisja danych
- Wiedzieć, jakie zastosowania ma moduł radio
- Umieć konwertować typy danych
- Znać podstawowe funkcje biblioteki radio
- Potrafić przesyłać i odbierać dane za pomocą fal radiowych na płytce Micro:bit w języku Python

Materiały pomocnicze:
- http://microbit.org/
- https://bbcmicrobitmicropython.readthedocs.io/
- https://github.com/MicrobitPolska/FunWithMicrobit

Pojęcia kluczowe:
→ radio → kanał radiowy → konfiguracja → transmisja danych

Czas realizacji: 45 min.

Metody pracy:
- wykład,
- dyskusja,
- ćwiczenia praktyczne przy komputerze,
- szukanie rozwiązań na postawiony problem
- prezentowanie efektów pracy.

Treści programowe:
Podstawa programowa kształcenia ogólnego dla szkół podstawowych – II etap edukacyjny – klasy VII-VIII, informatyka:
I. Rozumienie, analizowanie i rozwiązywanie problemów. Uczeń:
1) formułuje problem w postaci specyfikacji (czyli opisuje dane i wyniki) i wyróżnia kroki w algorytmicznym rozwiązywaniu problemów. Stosuje różne sposoby przedstawiania algorytmów, w tym w języku naturalnym, w postaci schematów blokowych, listy kroków;
3) przedstawia sposoby reprezentowania w komputerze wartości logicznych, liczb naturalnych (system binarny), znaków (kody ASCII) i tekstów;
4) rozwija znajomość algorytmów i wykonuje eksperymenty z algorytmami, korzystając z pomocy dydaktycznych lub dostępnego oprogramowania do demonstracji działania algorytmów;
5) prezentuje przykłady zastosowań informatyki w innych dziedzinach, w zakresie pojęć, obiektów oraz algorytmów.
II. Programowanie i rozwiązywanie problemów z wykorzystaniem komputera i innych urządzeń cyfrowych. Uczeń:
1) projektuje, tworzy i testuje programy w procesie rozwiązywania problemów. W programach stosuje: instrukcje wejścia/wyjścia, wyrażenia arytmetyczne i logiczne, instrukcje warunkowe, instrukcje iteracyjne, funkcje oraz zmienne i tablice;
2) projektuje, tworzy i testuje oprogramowanie sterujące robotem lub innym obiektem na ekranie lub w rzeczywistości;
4) zapisuje efekty swojej pracy w różnych formatach i przygotowuje wydruki;
III. Posługiwanie się komputerem, urządzeniami cyfrowymi i sieciami komputerowymi. Uczeń:
2) rozwija umiejętności korzystania z różnych urządzeń do tworzenia elektronicznych wersji tekstów, obrazów, dźwięków, filmów i animacji;
3) poprawnie posługuje się terminologią związaną z informatyką i technologią.
IV. Rozwijanie kompetencji społecznych. Uczeń:
1) bierze udział w różnych formach współpracy, jak: programowanie w parach lub w zespole, realizacja projektów, uczestnictwo w zorganizowanej grupie uczących się, projektuje, tworzy i prezentuje efekty wspólnej pracy;

Wprowadzenie w tematykę i integracja grupy (5 min.)

Nauczyciel rozmawia z uczniami na temat komunikacji bezprzewodowej. Pytam, jakie urządzenia odbierają informacje bezprzewodowo (np. radio FM), jakie wysyłają (nadajnik radiowy, satelitarny), a jakie wykonują obie te czynności (router bezprzewodowy, CB-radio, telefon komórkowy).

Część zasadnicza (35 min.)

Działanie programu będzie polegało na wysyłaniu tekstu, liczb lub obrazków z pierwszego urządzenia, a następnie odebraniu i wyświetleniu na drugim urządzeniu. Uczniowie powinni dobrać się w grupy i ustalić, która z nich będzie nadawać, a która odbierać sygnały.

Do tego celu skorzystamy z modułu radio i podstawowych funkcji jakie nam oferuje. W kodzie programu skorzystamy z następujących funkcji:

- **on()** - włącza radio na urządzeniu (komponent zwiększa zużycie prądu na płytce Micro:bit i wykorzystuje dodatkową pamięć);
- **off()** - wyłącza radio na urządzeniu (zmniejsza zużycie prądu i zwalnia pamięć);
- **config()** - konfiguruje parametry radia/transmisji (np. kanały);
- **send()** - wysyła wiadomość w postaci ciągu znaków;
- **receive()** - odbiera wiadomość w postaci ciągu znaków.

Nauczyciel powinien przydzielić każdej grupie unikatowy kanał (liczba z zakresu 0-83) w celu uniknięcia konfliktów w transmisji danych pomiędzy grupami. Do skonfigurowania kanału należy użyć funkcji **radio.config()** z argumentem **channel**.

Zaczynamy od prostego przykładu, tak aby zapoznać się z działaniem modułu radio - przesyłanie tekstu z jednego urządzenia i odbiór oraz wyświetlenie tekstu na drugim. Kod nadajnika będzie prezentował się następująco:

```
1  import radio
2  from microbit import *
3
4  radio.config(channel=1)
5
6  radio.on()
7
8  while True:
9      radio.send("Hello!")
```

Większość programu to realizacja pętli w ramach której cały czas wysyłamy napis **Hello!**

Z kolei kod odbiornika będzie wygląda następująco:

```
1  import radio
2  from microbit import *
3
4  radio.config(channel = 1)
5
6  radio.on()
7
8  while True:
9      text = radio.receive()
10     if text:
11         display.scroll(text)
```

Tutaj podobnie program będzie realizował przez cały czas pętlę, która będzie odbierała tekst. Jeśli odbiór tekstu się powiedzie zostanie on wyświetlony z wykorzystaniem metody **scroll** z modułu **display**. Cała komunikacja pomiędzy nadajnikiem a odbiornikiem w powyższych przykładach odbywa się na kanale pierwszym.

W kolejnym przykładzie będziemy wybierać liczby z wykorzystaniem przycisku i przesyłać je na drugie urządzenie. W tym celu będziemy konwertować je do postaci tekstu. Poniżej znajduje się kod nadajnika:

```
1  import radio
2  from microbit import *
3
4  radio.config(channel = 5) # konfiguracja radia, aby wykorzystywało kanał 5-ty
5
6  number = 1
7
8  while True:
9      if button_a.was_pressed():
10         if number < 9:
11             number += 1
12         else:
13             number = 1
14     elif button_b.was_pressed():
15         radio.on() # włączenie radia
16         radio.send(str(number)) # wysłanie liczby poprzez radio
17         radio.off() # wyłączenie radia
18     display.show(number)
```

Głównym elementem programu jest pętla, w której obsługujemy zdarzenia naciśnięcia przycisków A i B. Przycisk A służy do wyboru liczby, natomiast przycisk B służy do wysłania jej na drugie urządzenie. Zastosowana metoda **radio.send()** przyjmuje typ danych **str** (ciąg znaków), dlatego musimy wcześniej przekonwertować naszą liczbę na ten typ, używając do tego funkcji **str()**.

Kod odbiornika prezentuje się następująco:

```
1  import radio
2  from microbit import *
3
4  radio.config(channel = 5) # radio będzie wykorzystywać kanał 5.
5  radio.on() # włączenie radio
6
7  while True:
8      number = radio.receive() # nasłuchiwanie i odbiór danych
9      if number: # sprawdzenie czy coś zostało przesłane
10         display.show(number) # wyświetlenie przesłanej liczby
```

Podobnie jak w przypadku nadajnika główna część programu opiera się o pętle, w której cały czas próbujemy odbierać dane. Jeśli żadne dane nie zostały wysłane przez nadajnik, zmienna **number** przyjmuje wartość pustą **None**. Natomiast w momencie pojawienia się danych wyrażenie w instrukcji warunkowej **if** staje się prawdą i w efekcie na wyświetlaczu pokazuje się przesłana liczba.

Po udanym przetestowaniu wcześniejszych wersji programów przechodzimy do kolejnego etapu, w ramach którego będziemy wysyłać obrazki zamiast liczb. Do tego celu wykorzystamy między innymi poznaną na jednej z wcześniejszych lekcji listę obrazków. Poniżej znajduje się kod nadajnika:

Tutaj podobnie program będzie realizował przez cały czas pętlę, która będzie odbierała tekst. Jeśli odbiór tekstu się powiedzie zostanie on wyświetlony z wykorzystaniem metody **scroll** z modułu **display**. Cała komunikacja pomiędzy nadajnikiem a odbiornikiem w powyższych przykładach odbywa się na kanale pierwszym.

W kolejnym przykładzie będziemy wybierać liczby z wykorzystaniem przycisku i przesyłać je na drugie urządzenie. W tym celu będziemy konwertować je do postaci tekstu. Poniżej znajduje się kod nadajnika:

```
import radio
from microbit import *

radio.config(length = 42, channel = 5) # konfiguracja radia
index = 0
images_list = [Image.HAPPY, Image.SAD, Image.ANGRY, Image.ASLEEP,
Image.CONFUSED]

while True:
    if button_a.was_pressed():
        index += 1
        if index == len(images_list):
            index = 0
    elif button_b.was_pressed():
        radio.on() # włączenie radia
        radio.send(repr(images_list[index])) # wysłanie obrazka poprzez radio
        radio.off() # wyłączenie radia
    display.show(images_list[index])
```

Przy początkowej konfiguracji radia musimy zwiększyć maksymalną długość pojedynczej wiadomości w bajtach, aby móc wysyłać obrazki. W tym celu w funkcji **radio.config()** ustawiamy parametr **length**.

Funkcja **repr()** konwertuje obiekt obrazka do ciągu znakowego. Przykładowo dla **Image.HAPPY** ciąg ten prezentuje się w następujący sposób: **"Image('00000:09090:00000:90009:09990:')"**

Taka konwersja jest konieczna, żeby można było skorzystać z funkcji **send()**.

Modyfikujemy też nieznacznie kod odbiornika w porównaniu z poprzednią wersją:

```
import radio
from microbit import *

radio.config(length = 42, channel = 5)  # radio będzie wykorzystywać kanał 5-ty
radio.on()  # włączenie radio

while True:
    image = radio.receive() # nasłuchiwanie i odbiór danych
    if image: # sprawdzenie czy coś zostało przesłane
        image = eval(image)
        display.show(image) # wyświetlenie przesłanej liczby
```

Otrzymując obrazek w postaci ciągu znakowego po stronie odbiornika, konwertujemy go ponownie na obiekt typu **Image** przy pomocy funkcji **eval()**.

Podsumowanie (5 min.)

Pod koniec lekcji nauczyciel dyskutuje z uczniami, co było najtrudniejszym elementem zrealizowanych programów i komunikacji z użyciem modułu radio. Uczniowie prezentują efekty swojej pracy.

Alternatywa

Bardziej skomplikowana wersja programu zawiera połączony kod nadajnika i odbiornika, w wyniku czego otrzymujemy prosty czat obrazkowy (dwukierunkowa komunikacja).

```
1  import radio
2  from microbit import *
3
4  radio.config(length = 42, channel = 5) # konfiguracja radia
5  radio.on() # włączenie radio
6
7  index = 0
8  images_list = [Image.HAPPY, Image.SAD, Image.ANGRY, Image.ASLEEP,
9  Image.CONFUSED]
10 recv_mode = False # zmienna pomocnicza, określająca początkowy tryb - nadajnik
   lub odbiornik
11
12 while True:
13     if recv_mode: # jeżeli znajdujemy się w trybie odbiornika
14         data = radio.receive()
15         if data:
16             image = eval(data)
17             display.show(image)
18             sleep(2500)
19             display.show(" ")
20             sleep(500)
21             recv_mode = False
22     else: # jeżeli znajdujemy się w trybie nadajnika
23         if button_a.was_pressed():
24             index += 1
25             if index == len(images_list):
26                 index = 0
27         elif button_b.was_pressed():
28             radio.send(repr(images_list[index])) # wysłanie obrazka
29             recv_mode = True
30         display.show(images_list[index])
```

Istotnym dodatkiem powyższego programu, jest zmienna **recv_mode** , która określa czy urządzenie w danej chwili pełni rolę nadajnika lub odbiornika. Urządzenia na przemian zamieniają się rolami - w efekcie czat odbywa się w trybie turowym. Otrzymany obrazek znika po 2.5 sekundy na odbiorniku, który staje się nadajnikiem i umożliwia wybranie z przewijanej listy wysyłanego obrazka.