PRACOVNÍ LIST V-4

Co se naučíte

- Sériový přenos
- Propojit dva micro:bity rádiovou sítí
- Odeslání i příjem signálu

Co budete potřebovat

- PC s nainstalovaným editorem mu
- Propojovací USB kabel s micro USB koncovkou
- Micro:bit

A jděte na to …

Rozdělte se do dvojic a domluvte se kdo ve dvojici bude *Vysílač* a kdo *Přijímač*.

Vysílač odladí na Micro:bitu následující program:

```
1. from microbit import *
2. import radio
3. radio.on()
4. radio.config(channel=23)
5. while True:
6.     if button_a.is_pressed():
7.         radio.send("Zprava")
8.     sleep(1000)
9. radio.off()
```

Přijímač odladí následující:

```
1. from microbit import *
2. import radio
3. radio.on()
4. radio.config(channel=23)
5. while True:
6.     zprava = radio.receive()
7.     if (zprava):
8.         display.scroll(zprava)
9.         zprava = ""
10. radio.off()
```

Vyzkoušejte přenos signálu. K čemu slouží nastavení kanálu (channel)?

Vyměňte si role a zopakujte si zadání v opačných pozicích.

Pohovořte si o možné bezpečnosti přenosu.

PRŮVODCE TEORIÍ

Počítačové sítě

Než se pustíme do práce se sítí uvedeme si pár teoretických definic, abychom později lépe porozuměli tomu, co stavíme a programujeme.

Počítačové sítě dělíme dle přenosového média na:

- Drátové
- Bezdrátové
    - Wi-Fi
    - Bluetooth
    - Radio
    - Infra

My budeme micro:bity propojovat jak *drátově*, tak *bezdrátově* pomocí radia. Micro:bit sice obsahuje i Bluetooth přijímač a vysílač, ale jak bude dále vysvětleno v MicroPythonu jej nelze použít.

Dle směru vysílání dělíme sítě na:

Simplexové – vysílání jde pouze jedním směrem, na jedné straně se nachází vysílač (sender nebo transmitter) a na druhé přijímač (receiver).

Half duplex (poloviční duplex) – vysílání může jít oběma směry, ale v daném okamžiku pouze jedním směrem.

Duplex – vysílání může jít v daném okamžiku oběma směry současně

Síťový protokol – soubor předem domluvených pravidel, kterými se řídí daný síťový přenos.

Drátový přenos

Pro následující příklad budeme potřebovat dva micro:bity. Oba propojíme pomocí dvou kabelů tak, že propojíme vzájemně piny 1 a piny 2 na obou micro:bitech. (Pin1 na prvním micro:bitu s pinem1 na druhém micro:bitu a stejně i pin2.) Opět můžeme s výhodou použít kabely s „krokodýlky“ na koncích.

Je třeba stanovit, který micro:bit bude „Vysílač“ a který „Přijímač“. Jedná se tedy o simplexový přenos. Funkce programů je následující na prvním micro:bitu stiskneme tlačítko A nebo B a na druhém micro:bitu se vzápětí rozsvítí symbol A nebo B. Jedná se tedy o binární stav, jeden ze symbolů může reprezentovat jedničku (pravdu – true) a druhý nulu (nepravdu – false). Pomocí síťového protokolu lze domluvit, co v daném případě, který symbol znamená a o jakou informaci se jedná.

Na micro:bitu „Vysílač“ nahrajte a odlaďte následující program:

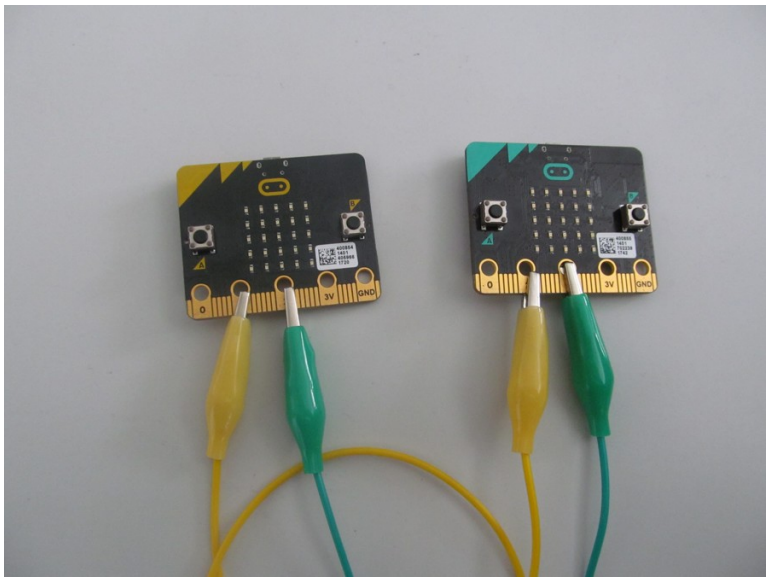
```
1. from microbit import *
2. while True:
3.     if button_a.is_pressed():
4.         display.show("A")
5.         pin1.write_digital(1)
6.     else:
7.         pin1.write_digital(0)
8.     if button_b.is_pressed():
9.         display.show("B")
10.           pin2.write_digital(1)
11.       else:
12.           pin2.write_digital(0)
13.       display.clear()
14.   sleep(10)
```

Program v nekonečném cyklu sleduje, zda bylo stisknuté tlačítko a, nebo tlačítko b a podle toho vysílá jedničku na daném propojení. Pro kontrolu zobrazuje rovněž písmeno dle stisknutého tlačítka.

Na micro:bitu „Přijímač“ nahrajte a odlaďte následující program:

```
1. from microbit import *
15. while True:
16.     if pin1.read_digital():
17.         display.show("A")
18.     elif pin2.read_digital():
19.         display.show("B")
20.     sleep(1000)
21.     display.clear()
```

Tento program v nekonečné smyčce kontroluje, zda je na některém z připojených vodičů signál a pak zobrazuje odpovídající písmeno. Je třeba si také uvědomit, že v daném případě nelze říci co se stane, když bude signál na obou vodičích. To jaké písmeno (zpráva) se zobrazí záleží na naprogramování, tedy na předchozí domluvě. Tomu se říká *Síťový protokol*.



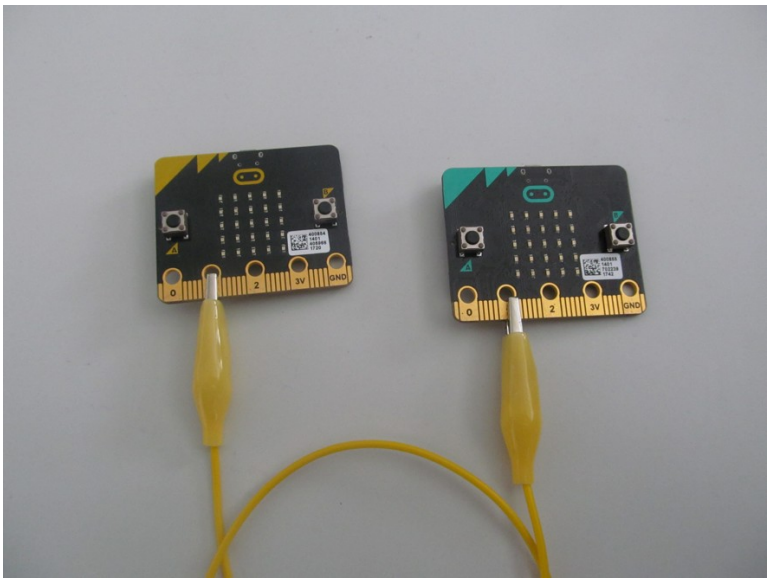
Tomuto případu, kdy signál putuje po více kabelech se říká *paralelní přenos* – signály posíláme vedle sebe. Bylo by možné využít tohoto zapojení k přenosu 4 znaků, s následujícími možnostmi:

1. žádný signál
2. signál na prvním vodiči
3. signál na druhém vodiči
4. signál na obou vodičích

Kolik znaků bychom mohli přenést při maximálním možném počtu vodičů, který umožňuje micro:bit? Odpověď závisí na tom, zda použijeme tři vodiče v běžném režimu (pomocí krokodýlků) anebo až 17 vodičů, pokud použijeme datový shield. Vždy je to však 2^n.

Pokud chceme propojit micro:bity pouze jedním kabelem, a i pak přenášet jiný signál, než zapnuto vypnuto, je nutno se domluvit na nějakém protokolu. V následujícím příkladu odlišujeme přenesený signál dle jeho délky – 500 ms nebo 1500 ms. Měříme jeho délku pomocí funkce running.time() a pokud je signál kratší než jedna sekunda, považujeme jej za první stav a pokud je delší než jedna sekunda za druhý stav. Takovémuto způsobu přenosu se říká *sériový* – signály posíláme za sebou.

Micro:bity propojíme pouze jedním kabelem mezi piny 1.



Na micro:bit „Vysílač nahrajeme následující program:

```
1. from microbit import *
2. while True:
3.     if button_a.is_pressed():
4.         display.show("A")
5.         pin1.write_digital(1)
6.         sleep(500)
7.         pin1.write_digital(0)
8.     if button_b.is_pressed():
9.         display.show("B")
10.           pin1.write_digital(1)
11.           sleep(2000)
12.           pin1.write_digital(0)
13.  display.clear()
```

Program v nekonečné smyčce hlídá stisk kláves A a B a při stisku vyšle signál odpovídající délky a pro kontrolu zobrazí i kód stisknuté klávesy. Na micro:bit „Přijímač“ nahrajeme následující program:

```
1. from microbit import *
2. while True:
3.     if pin1.read_digital():
4.         start = running_time()
5.         while pin1.read_digital():
6.             pass
7.         konec = running_time()
8.         cas = konec - start
9.         if cas < 1000:
10.             display.show("A")
11.         else:
12.             display.show("B")
13.         sleep(1000)
14.         display.clear()
```

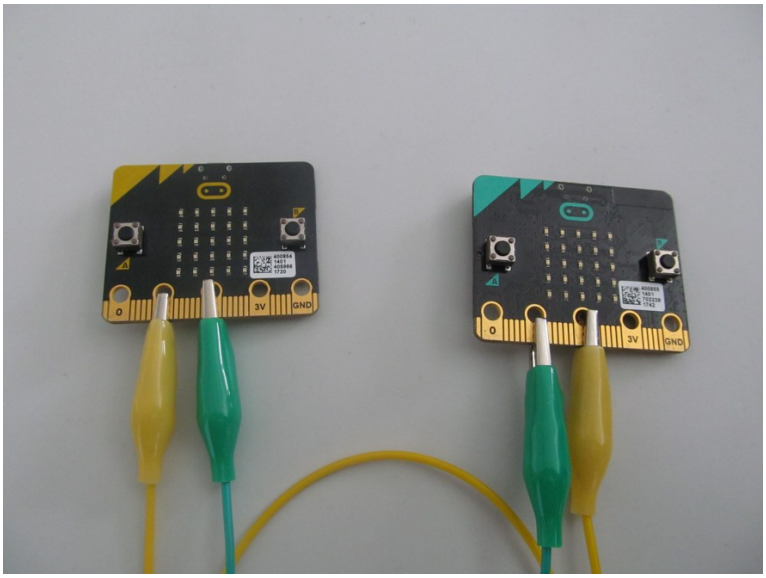
Program v nekonečné smyčce hlídá, zda se objeví signál na pinu1. Pokud ano zaznamená si jeho čas. Funkce running.time() vrací čas v milisekundách od spuštění micro:bitu. Nyní se čeká dokud je na pinu1 signál a po jeho ukončení opět změříme čas. Spočítáme dobu signálu odečtením začátku od konce. Je-li signál kratší než jedna sekunda považujeme jej za první typ znaku – A. Je-li delší pak za B.

Tímto způsobem si můžeme přenášet binární signál, namísto znaků A a B, můžeme použít 1 a 0 (nebo naopak). Tímto způsobem je možné například přenášet ASCII kód, když si řekneme, že osm za sebou přenesených znaků je kód jednoho ASCII znaku. Lze takto například přenášet i Morseovu abecedu, pokud si například řekneme, že kratší znak je tečka, delší znak je čárka.

Zájemci si mohou zkusit obě tyto úlohy řešit. Vyřešení úlohy s přenosem kódu Morseovy abecedy naleznete na internetových stránkách MicroPythonu.

Nyní opět změníme zapojení. Propojíme micro:bity tak, že kabel bude na jednom micro:bitu připojen na pin1 a na druhém na pin2. S druhým kabelem to uděláme naopak. Micro:bity budou tedy zapojeny do kříže na pinech 1 a 2. Oba micro:bity budou nyní současně „Přijímač“ i „Vysílač“ na oba tedy nahrajeme stejný kód.

```
1. from microbit import *
2. while True:
3.     if pin1.read_digital():
4.         display.show(Image.HAPPY)
5.     else:
6.         display.clear()
7.     if button_a.is_pressed():
8.         pin2.write_digital(1)
9.     else:
10.         pin2.write_digital(0)
11.     sleep(100)
```



Význam kódu je velice jednoduchý. Program v nekonečné smyčce provádí následující. Pokud je na pinu 1 signál, zobrazí se smajlík. Pokud je stisknutá klávesa A vyšleme signál na pin 2. To samé se děje i na druhé straně. Jedná se o *duplexní signál*, současně přenášíme informaci oběma směry.

Možná vás nyní napadlo, že by mohl stačit jen jeden kabel. V tom případě musíme řešit následující dva problémy:

1. Pokud by micro:bit vyslal signál na vodič, současně by se na něm při testu signálu na témže vodiči načetla jednička a měl by za to, že signál i přijímá. Této situaci se dá zabránit tak, že v případě vysílání signálu se netestuje stav signálu na vodiči.
2. Další problém by vznikl by pokud oba micro:bity vysílaly současně. Takovémuto případu se říká *kolize*. Řešením je, že micro:bit před vysláním signálu nejprve ověří, zda na kabelu již není signál a pak teprve začne vysílat. Je třeba si uvědomit, že to není dokonalé řešení, oba micro:bity mohou testovat a začít vysílat ve stejném okamžiku. Je to málo pravděpodobné, ale ne zcela vyloučené. I tento případ má řešení, po začátku vysílání micro:bit po náhodné době na malou chvilku přeruší signál a ověří zda na vodiči není signál z druhé strany a pak buď počká nebo vysílá dál. Doba po, které se to testuje, nesmí být vždy stejná, aby opět obě strany nedělaly totéž.

Takovémuto přenosu jak již víme by se říkal *half duplex*. Můžete si jej zkusit naprogramovat, včetně řešení kolize. Toto je zjednodušení způsobu jakým je přenášen internet po běžných ethernetových kabfrom microbit import *

```
1. while True:
2.     if button_a.is_pressed():
3.         display.show("A")
4.         pin1.write_digital(1)
5.         sleep(500)
6.         pin1.write_digital(0)
7.     if button_b.is_pressed():
8.         display.show("B")
9.         pin1.write_digital(1)
10.        sleep(2000)
11.     pin1.write_digital(0)
12.     display.clear()
```

Bluetooth přenos

Ačkoliv Micro:bit obsahuje přijímač i vysílač bluetooth signálu, nelze jej v MicroPythonu využít. Na vině je to, že do paměti micro:bitu nelze současně umístit překladač MicroPythonu i kód pro obsluhu bluetooth kvůli velikosti.

Problémem je i to, že pokud jsme na micro:bitu pracovali s MicroPythonem, nelze jej pomocí bluetooth spárovat s jiným zařízením. Jediné možné řešení je nahrát na micro:bit libovolný program vytvořený grafickým rozhraním na stránkách microbit.org. Pak již micro:bit můžeme spárovat s jiným zařízením prostřednictvím bluetooth.

Rádiový přenos

Velice zajímavou možností jak mohou spolu dva Micro:boty komunikovat je bezdrátový rádio přenos. Je možné naladit celkem 84 kanálů označených 0 až 83. Jedná se o frekvenci 2400 MHz (odpovídá kanálu 0), každý kanál má rozsah cca. 1 MHz.

Práce s tímto přenosem je velmi jednoduchá, MicroPython obsahuje knihovnu, která má v sobě metody umožňující přímo přenos textového řetězce (nebo čísla). Ukázka je v následujícím příkladě. Na straně odesílatele:

```
1. from microbit import *
2. import radio
3. radio.on()
4. radio.config(channel=23)
5. while True:
6.     if button_a.is_pressed():
7.         radio.send("Zprava")
8.     sleep(1000)
9. radio.off()
```

A na straně příjemce:

```
1. from microbit import *
2. import radio
3. radio.on()
4. radio.config(channel=23)
5. while True:
6.     zprava = radio.receive()
7.     if (zprava):
8.         display.scroll(zprava)
9.         zprava = ""
10. radio.off()
```

Parametr channel je nepovinný, pokud jej neuvedete, pak předvolený je kanál 7. Na druhou stranu, pokud si má spolu povídat větší množství micro:bitů, je nutné, aby ty které k sobě patří používaly nezávislý kanál a nepletly se tak ostatním.

Na tomto příkladě je rovněž vidět potenciální nebezpečí. Pokud útočník ví na kterém kanále si naše micro:bity povídají, může je buď odposlouchávat anebo podstrkovat vlastní zprávy. Jedná se o typ útoku Man in the middle.

Možné použití této technologie se nabízí v následujících možnostech: