LEGO MindStorm / LEGO Dacta

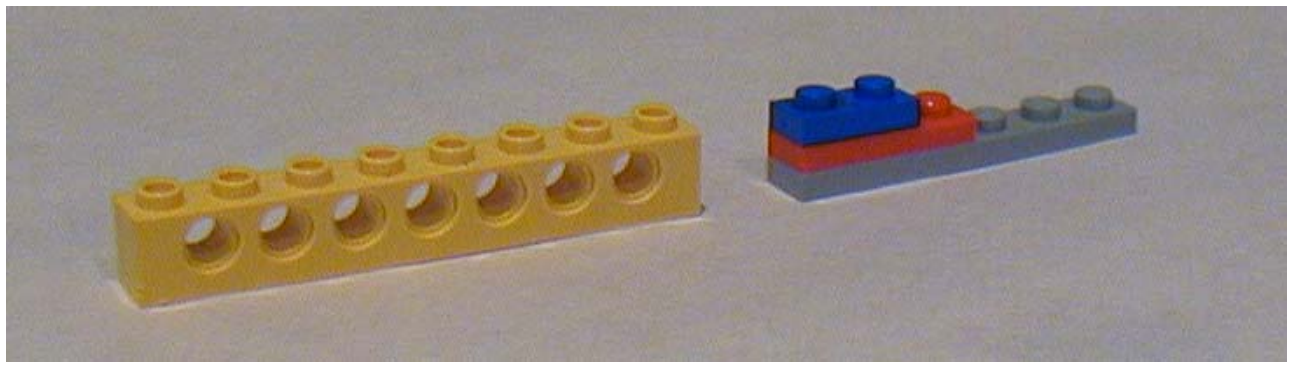
František Mráz
KSVI MFF UK

LEGO MindStorm/Dacta

- Jednoduchosť a rýchlosť stavby robota
- LEGO je čisté
- cenovo dostupné cca. 9200,- s DPH (>800 dielikov, bez software)
  - Souprava obsahuje 826 dílů včetně dvou motorů s převodem, jedné světelné kostky, dvou tlakových a dvou světelných senzorů. Přiložené náměty činností neobsahují informace o programování. Souprava je uložena v kontejneru s mini kazetou. Součástí je RCX LEGO kostka a IR věž (9793-COM, 9794-USB), neobsahuje software. Určena je pro 4 žáky starší jedenácti let ke skupinové práci na projektu.
- Efektívne - dá sa postaviť mnoho modelov
- bezpečné

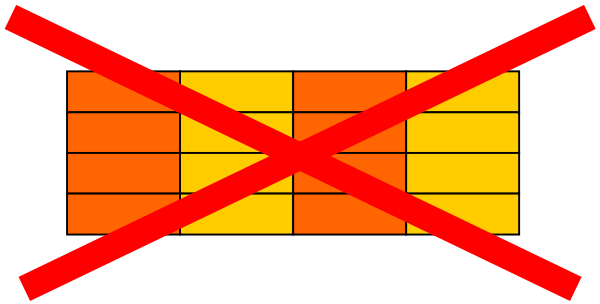
Geometria

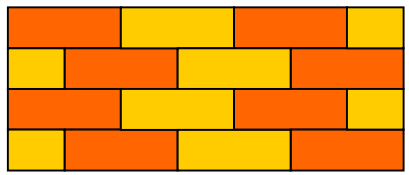
- 3 pláty = výška 1 tehly



- tehlička s 1 výstupkom 5/16” x 5/16” x 3/8” (bez výstupku výšky 1/16”)
- základná tehlička

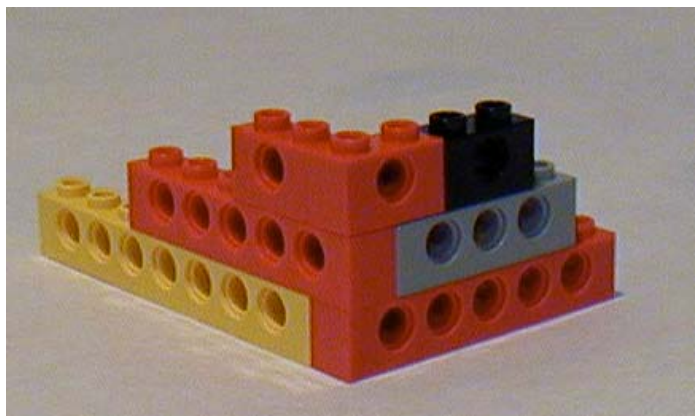
Chybné konštrukcie



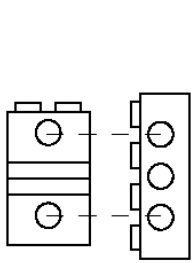


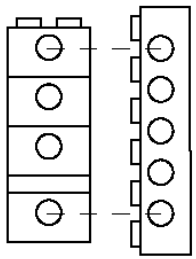
- zle
- iba trocha lepšie

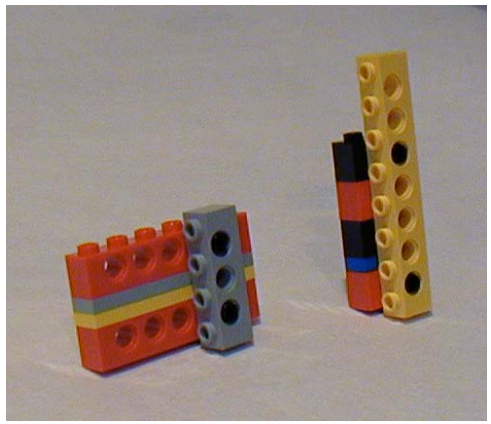
- ani toto príliš nevydrží



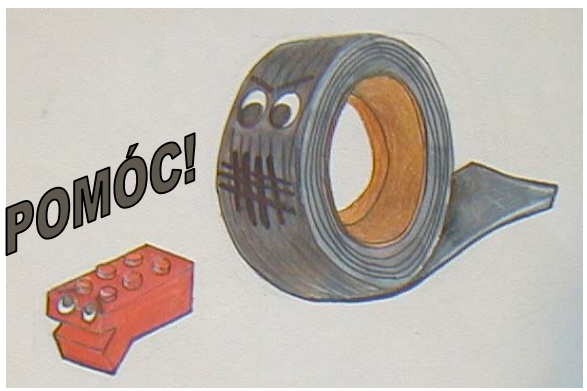
Správne





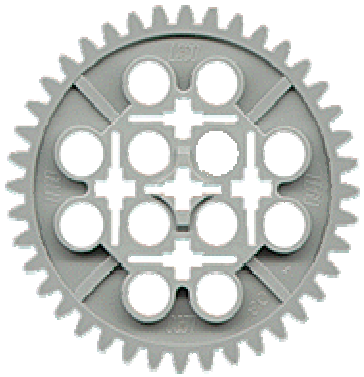


LEGO kocky nikdy nelepiť!



Pohon - hnacie súkolia

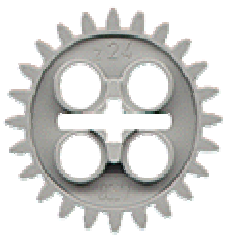
8z



40z

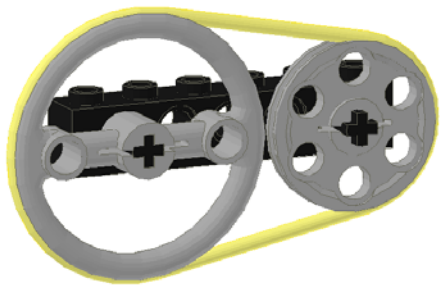
12z
zošikmené

16z



24z

1z slimák

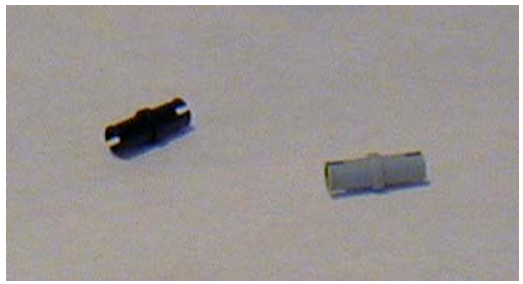


kladky

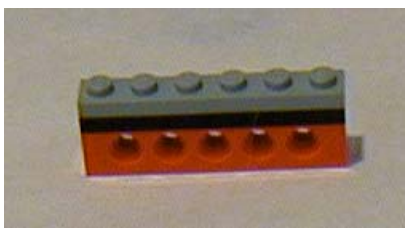
24z
veniec

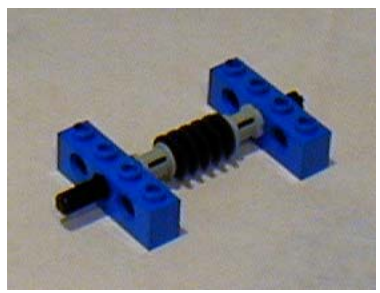
Spojovacie kolíky, prevodovka

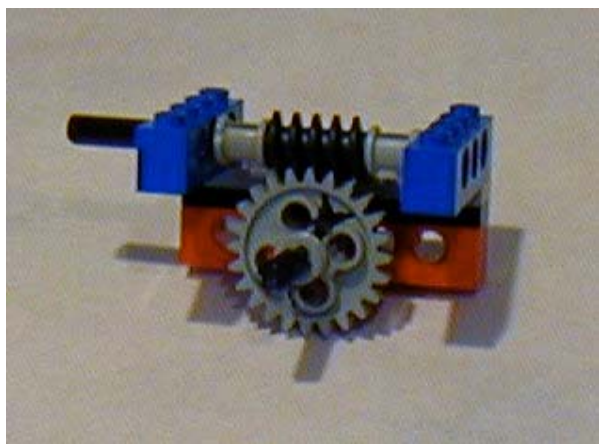
- Čierne na pevné spoje
- šedé na otočné spoje



- závitovkové koleso - slimák - 1 zub/otáčka
- konštrukcia prevodovky 24:1, spätný prevod nefunguje

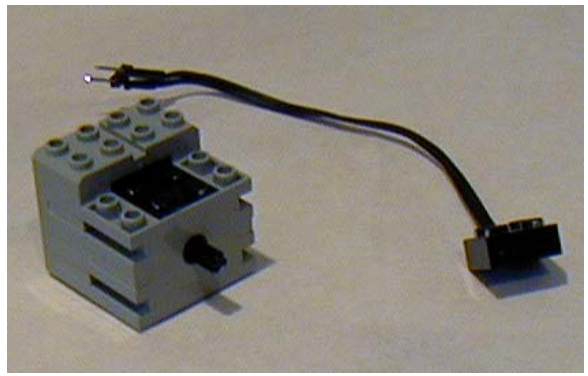




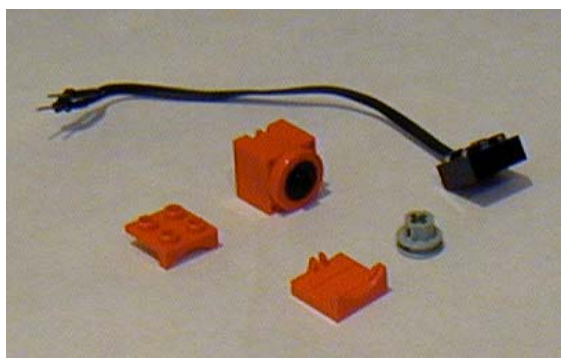


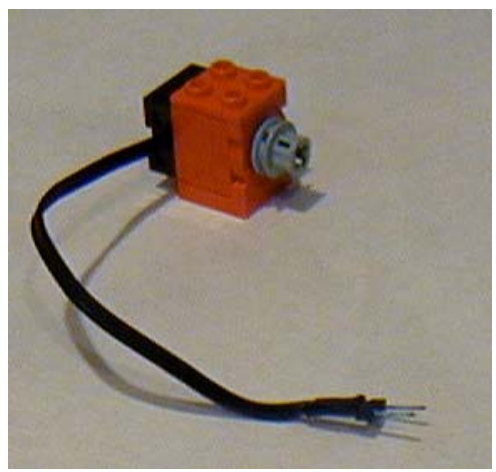
Motory

- 9V motor s prevodom
  - 10 - 250 mA
  - 200 - 350 ot./min

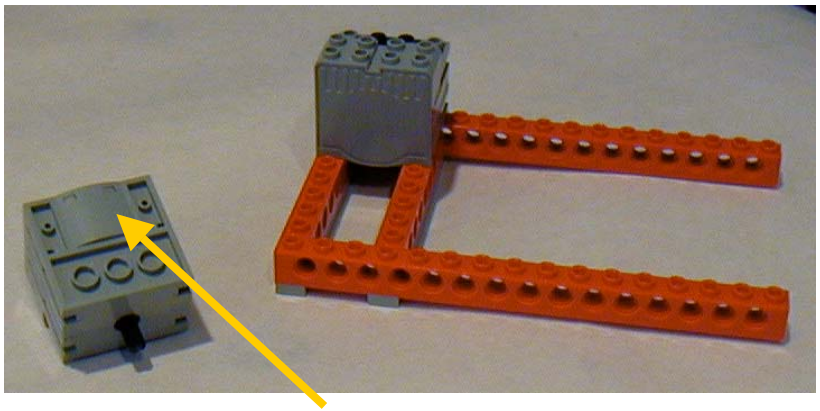


- 9V mikromotor
  - 5 - 90 mA
  - 20 - 30 ot./min (nezaťažený)

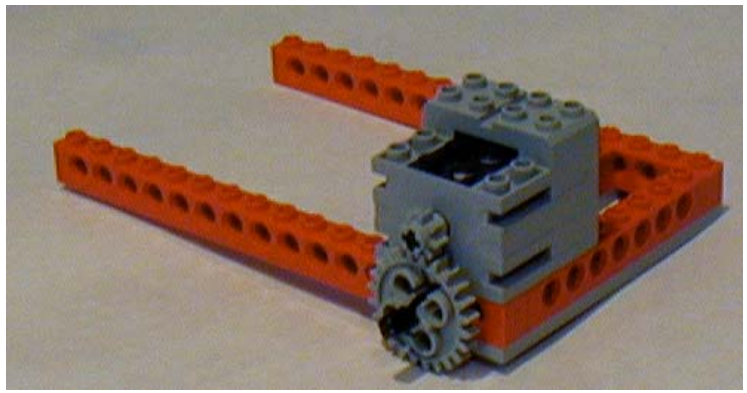


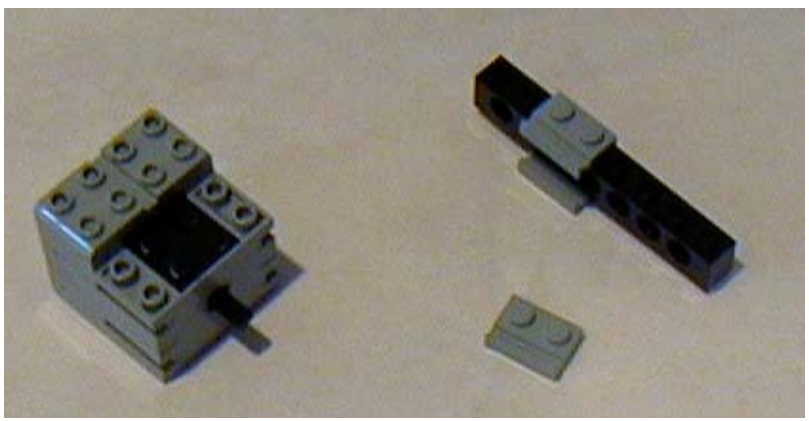


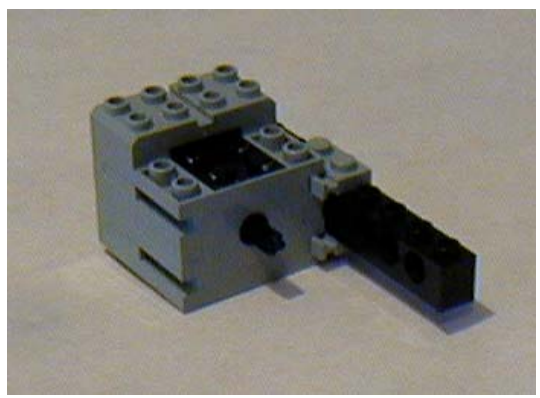
Zabudovanie motorov



výstupok



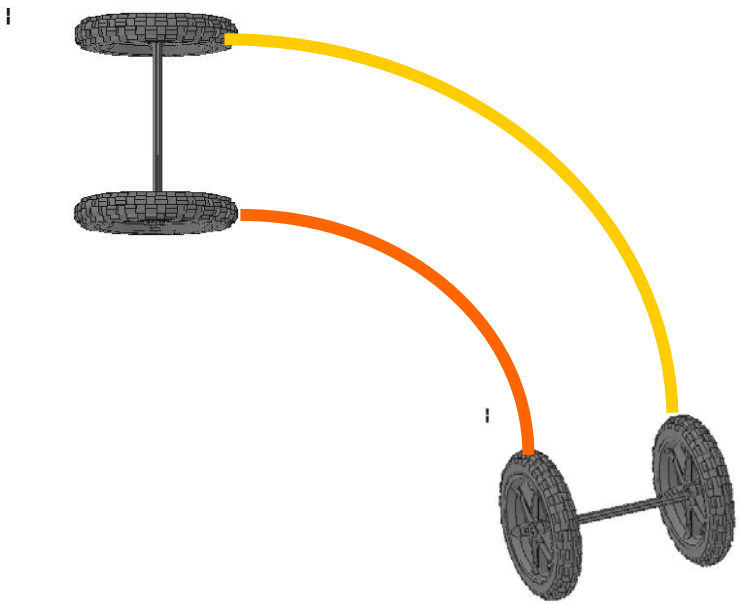


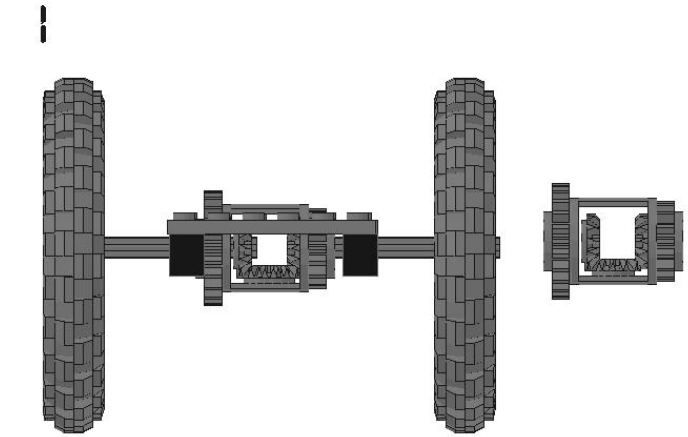


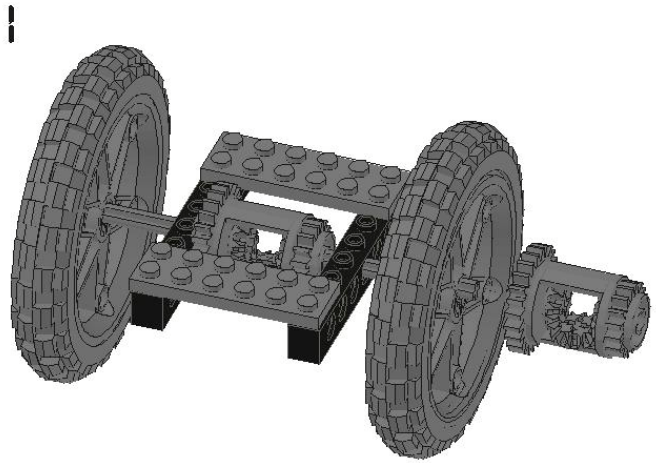
veľmi pevné

Diferenciána náprava

- Náprava s diferenciálom

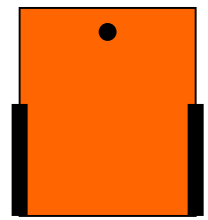




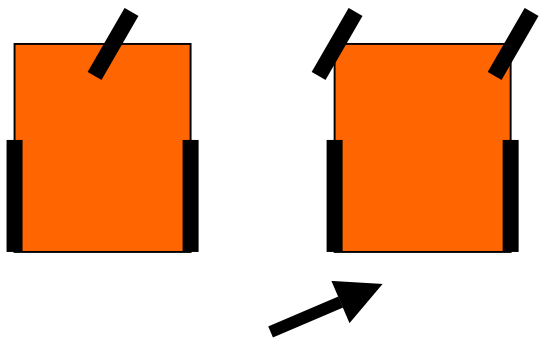


Podvozok a jeho riadenie

- Diferenciálny pohon - na jednej náprave sú dve nezávisle poháňané kola, ďalšie kolo iba opora (niekedy iba trn); umožňuje otočenie na mieste
  - 2 motory
  - žiadne dva motory nie sú rovnaké !
    - problém s držaním priameho smeru



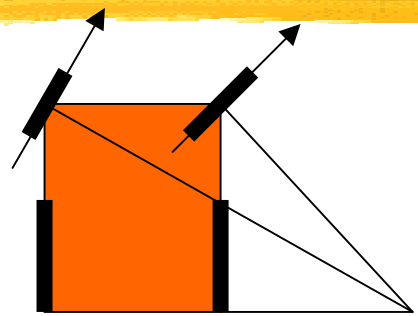
- jedna náprava riadená
  - druhá vyžaduje diferenciál



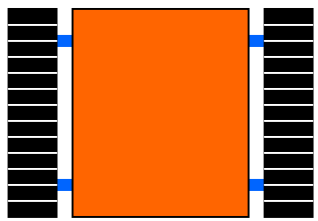
Preklzovanie, kolá by mali byť natočené rôzne

Podvozok a jeho riadenie

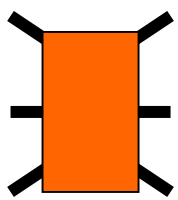
- Riešenie - Ackermanovo riadenie



- pásový pohon - preklzovanie nutné

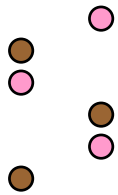


- iný pohon - kráčajúci robot
  - lepšia priechodnosť
  - zložitejšie riadenie
  - počet nôh 8,6,4,2



Typy chôdze

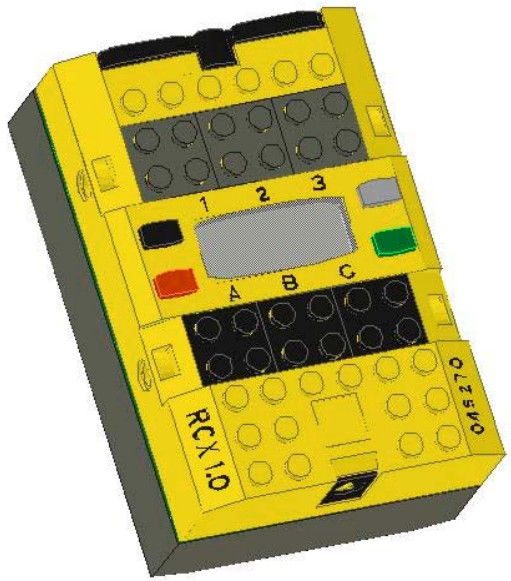
- Statická
  - trojkrok



  - štvorkrok (8-nohé), vlnový krok
- Dynamická
  - krok (mimochod)
  - klus

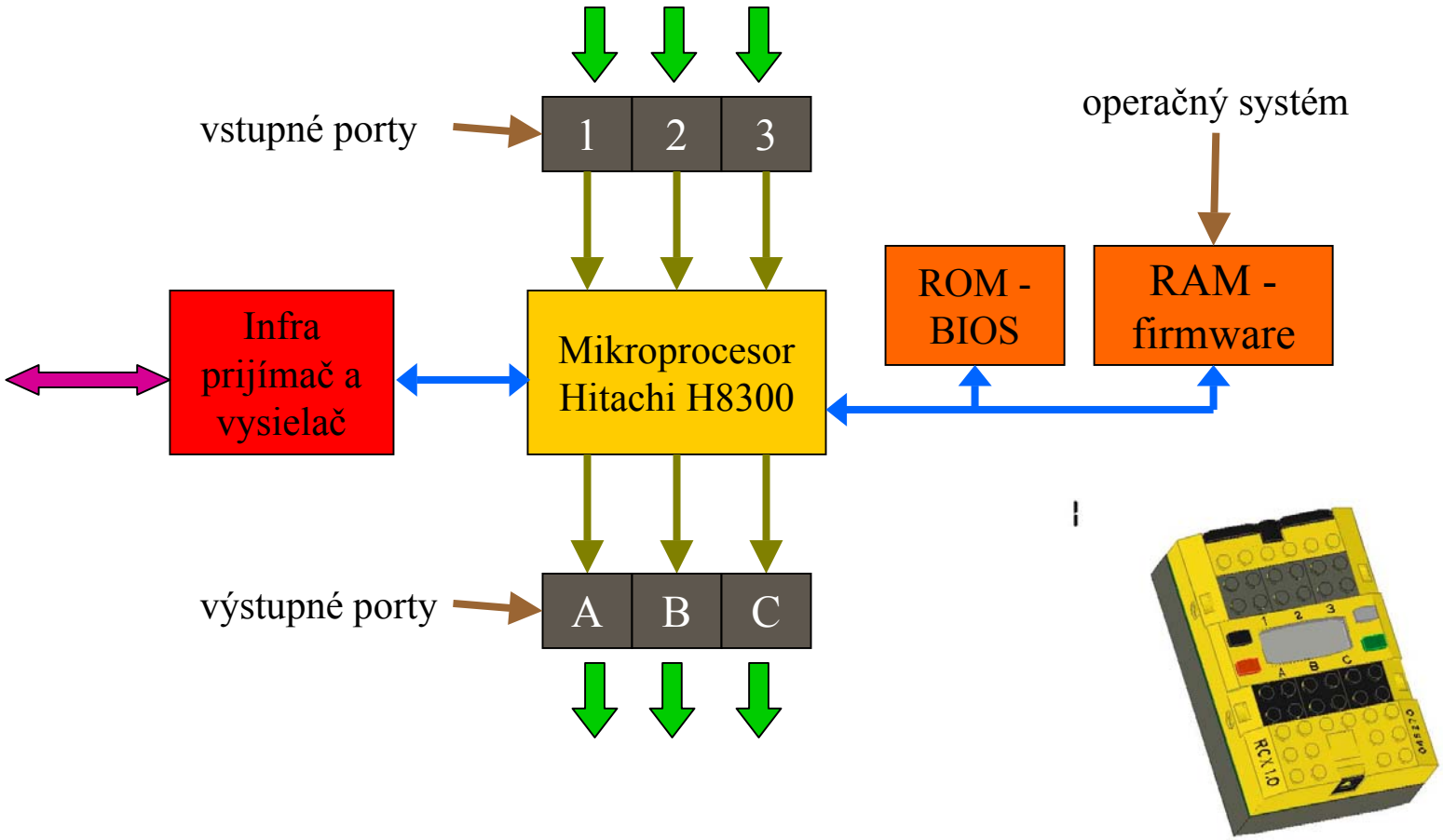
Riadiaci počítač - RCX kocka

- Veľmi obmedzený počítač
  - malý displej
  - tri vstupné porty
  - tri výstupné porty
  - 4 tlačítka
  - infračervené rozhranie
  - konektor na pripojenie adaptéra pre napájanie zo siete



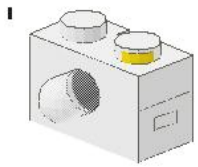
  - napájanie - 6 ceruzkových batérií, alebo akumulátorov
  - programovnie sa robí na PC (Mac), program sa pomocou infračervenej veže (IR Tower) pripojenej k PC pošle do RCX a tam sa vykonáva

Schéma RCX kocky

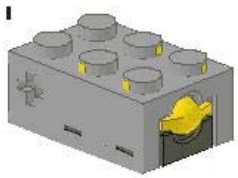


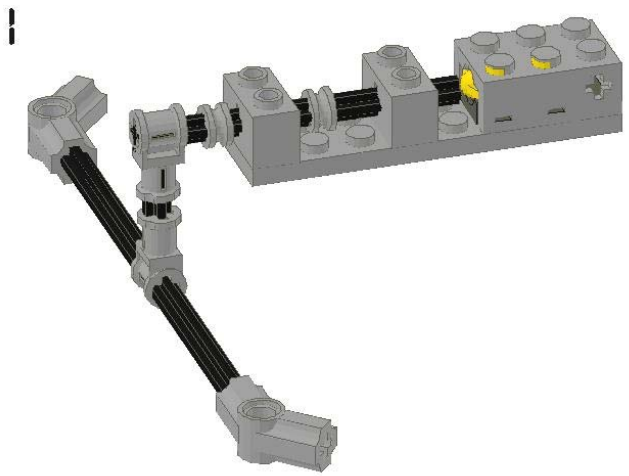
Výstupy / vstupy

- Výstupy
  - Motory
  - lampička - nastavenie intenzity svetla
  - reproduktor



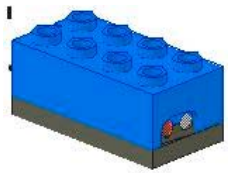
- vstupy - senzory
  - dotykový senzor
    - pasívny, áno/nie
    - nárazníky
    - detekcia polohy
    - počítanie otáčok



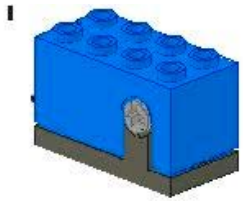


Senzory - pokračovanie

- svetelný senzor
  - meria intenzitu svetla
    - pasívne - okolité svetlo
    - aktívne - odrazené svetlo
    - 0 - 100 %; 0 - 1023



- rotačný senzor
  - natočenie - 16 hodnôt na 1 otáčku (22,5°)
  - zvýšenie presnosti - prevody



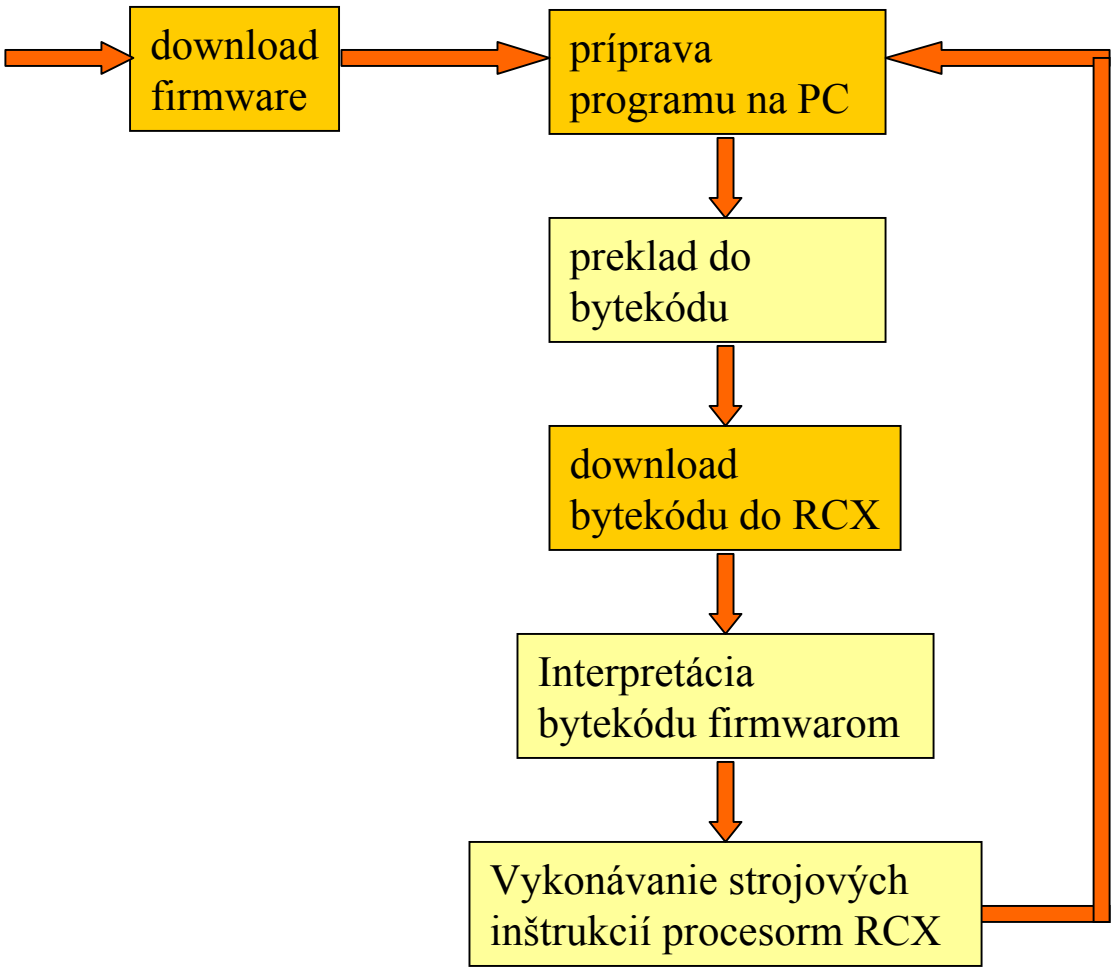
- kamera
  - cez PC
  - tzv.virtuálne senzory
    - napr. intenzita červenej v danom bode

Ďalšie senzory

- Teplotné čidlo
- vhkostné čidlo
- meranie kyslosti - ph

- infračervené oko - pre komunikáciu s PC prostredníctvom infračervenej veže - dá sa použiť jako radar
  - vysiela a odrazené infrasvetlo sa detekuje svetelným senzorom
  - prijíma signál od veže, alebo iného robota

Schéma práce s RCX kockou



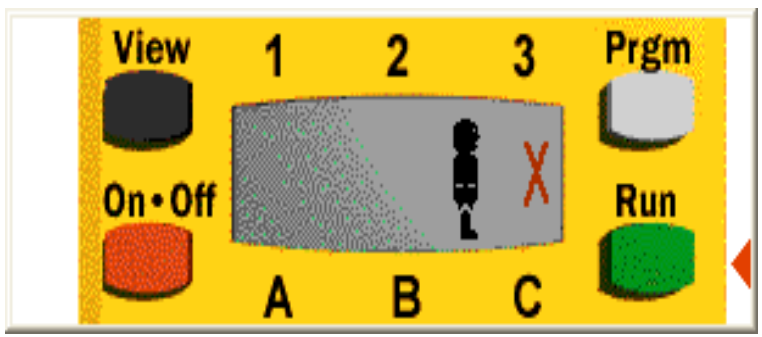
Programovanie robotov z LEGO stavebníc

- od firmy LEGO
  - Robolab
  - SDK - obsahuje assembler, moduly pre Visual Basic, ...
- NQC - Not Quite C
- pri použití alternatívneho firmware **brickOS**
  - gcc - C++, Pascal, …
- ďalšie
  - leJOs - Java
  - forth, ...

- **DOPORUČENIE: brickOS + gcc**
  - http://brickos.sourceforge.net/index.html

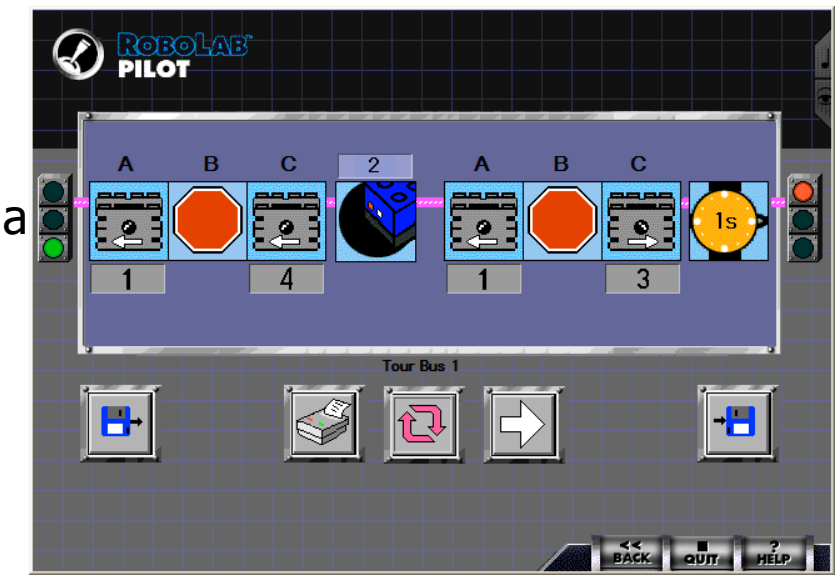
Robolab

- Administrator
  - nastavenie portu, IR veže, ...
- Programmer
  - pilot
    - pre začiatočníkov, iba postupnosť príkazov
    - 4 úrovne, 1-3 v programoch sa dajú meniť iba parametre
  - **Inventor**
- Investigator



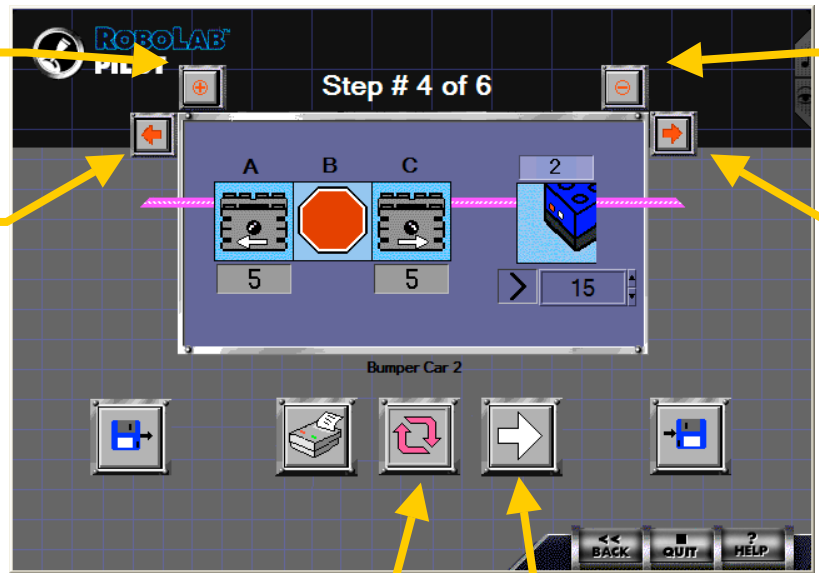
Pilot

- úroveň 1
  - port A - motor alebo lampička
  - nastavenie doby behu
- úroveň 2
  - porty A,C, nastavenie úrovne výstupu
  - nastavenie doby behu alebo stlačnie/uvoľnenie dotykového senzoru na niektorom porte
- úroveň 3
  - porty A,B,C, 2 kroky
  - možnosť nepretržitého opakovania celého programu
  - svetelný senzor - svetlo/tma



Pilot

- úroveň 4
  - neobmedzený počet krokov



pridať krok

predchádzajúci krok

vymazať krok

nasledujúci krok

opakovať

download

Inventor

- Verzia systému LabView
- programy - virtual instruments prípona .VI

NQC - Not Quite C (Dave Baum)

- Jazyk podobný C
  - začiatok programu
  - { a } ohraničujú blok
  - `OnFwd()` - spustiť motor dopredu
  - `Wait()` - počkať daný počet
  - stotín sekundy
  - `OnRev()` - změna otáčania
  - `Off()` - vypnutie výstupov
  - príkazy sú ukončené stredníkom (`;`)
  - `OUT_A, OUT_B, OUT_C` - 3 výstupy

```
task main()
{
  OnFwd(OUT_A);
  OnFwd(OUT_C);
  Wait(400);
  OnRev(OUT_A+OUT_C);
  Wait(400);
  Off(OUT_A+OUT_C);
}
```

- Vývojové prostredie - Bricx Command Center

Nastavenie atribútov

- Atribúty
  - spustenie/zastavenie: `OUT_OFF, OUT_ON, OUT_FLOAT`
    - `SetOutput(OUT_A, OUT_FLOAT);` - plynulé zastavenie motora, nie zabrzdenie
    - `Float(OUT_A); On(OUT_B); Off(OUT_C);`
  - smer `Direction: OUT_FWD, OUT_REV, OUT_TOGLE`
    - SetDirection(OUT_B, OUT_FWD);
    - `Fwd(OUT_C);`
  - úroveň `SetPower ("výstupy","rýchlosť");`
    - rýchlosť 0-7 (7 najrýchlejšie, pri 0 nestojí), závisí na záťaži
  - `OnFwd(P)` znamená to isté ako

    `SetOutput(P,OUT_ON); SetDirection(P,OUT_FWD);`
  - `OnFor("výstupy","čas (1/100 s)");`

Zatočenie

- komentáre `/* dlhy komentar */`

  `// komentar do konca riadku`
- `#define`
- zložený príkaz `{ "príkaz"; "príkaz"; … }`
- zopakovať 4 krát - jazda do štvorca

  `repeat(4) "príkaz";`

  viz. Štvorec.nqc

```
/* zatoc doprava */

#define MOVE_TIME 100
#define TURN_TIME 40

task main()
{
  OnFwd(OUT_A+OUT_C);
  Wait(MOVE_TIME);
  OnRev(OUT_C);
  Wait(TURN_TIME);
  Off(OUT_A+OUT_C);
}
```

Premenné

- Iba celočíselné; max. 32 premenných
- deklarácia

```
int a,b,c;
```

- aritmetické operácie

```
a = 1; b = 2; c=3;
a = b + c;
a = a + 2;  a += 2;
b = a / 2;  //celočís. delenie
c = a * b;  //násobenie
a = b - a * (b + 3);
a = Random(40)  //náhodné
          // číslo 0-40
Wait(10+Random(10));
```

```
#define TURN_TIME 40
int move_time;
task main()
{
  move_time = 50;
  repeat (20)
  {
    OnFwd(OUT_A+OUT_C);
    Wait(move_time);
    OnRev(OUT_C);
    Wait(TURN_TIME);
    move_time += 5;
  }
  Off(OUT_A+OUT_C);
}
```

Cykly

- until (“podmienka")
  “príkaz“
- until (“podmienka");
  // prázdne telo
- **while (“podmienka")**
  “príkaz“
- **do**
  “príkaz“
  while (“podmienka");

```
#define MOVE_TIME   100
#define TURN_TIME    40

task main()
{
  while(true)
  {
    OnFwd(OUT_A+OUT_C);
    Wait(MOVE_TIME);
    if (Random(1) == 0)
      { OnRev(OUT_C); }
    else
      { OnRev(OUT_A); }
    Wait(TURN_TIME);
  }
}
```

Podmienky - logické výrazy

- `True` *//vždy platí*
- `false` *//nikdy neplatí*
- `Random(1) == 0` *//test na rovnosť*
  - `==` *//rovná sa*
  - `!=` *//nerovná sa*
  - `<` *//menší*
  - `<=` *//menší alebo rovný*
  - `>` *//väčší*
  - `>=` *//väčší alebo rovný*
- `(a==3)`
- `(b>=5) && (b<10)` *//a zároveň*
- `(b<5) || (b>=10)` *//alebo*
- `!((b>=5) && (b<10))` *//negácia*

Senzory - čakanie na senzor

```
task main()
{
  SetSensor(SENSOR_1,SENSOR_TOUCH);
  OnFwd(OUT_A+OUT_C);
  until (SENSOR_1 == 1);
  Off(OUT_A+OUT_C);
}
```

- Senzory sa označujú: `SENSOR_1, SENSOR_2, SENSOR_3`
- vždy je třeba nastaviť typy senzorov
  - `SetSensor(SENSOR_1,SENSOR_TOUCH);`
- iné typy senzorov `SENSOR_LIGHT, SENZOR_ROTATION`

Sledovanie čiary

- Iba konvexná krivka
- v smere hodinových ručičiek
- správny prah sa dá zistiť použitím tlačítka view

```
#define THRESHOLD 40

task main()
{
  SetSensor(SENSOR_2,SENSOR_LIGHT);
  OnFwd(OUT_A+OUT_C);
  while (true)
  {
    if (SENSOR_2 > THRESHOLD)
    {
      OnRev(OUT_C);
      until (SENSOR_2 <= THRESHOLD);
      OnFwd(OUT_A+OUT_C);
    }
  }
}
```

Úloha - task

- Jeden program môže mať až 10 úloh, ktoré môžu bežať paralelne!
- pri štarte programu sa vždy spustí úloha **`main`**
- ďalšie úlohy sa spúšťajú príkazom

```
start „úloha“
```

- zastavenie úlohy
  - tým, že skončí
  - príkazom

```
stop „úloha“
```

- **príklad** **`Štvorec_2.nqc`**
  - **`main`** - nastaví typ senzora, spustí **`move_square`** **a** **`check_sensors`**
  - **`move_square`** **-** jazda do štvorca
  - **`check_sensors`** - ak dotykový senzor dá 1, zastaví **`move_square`**, vyhne, spustí **`move_square`**

Podprogramy

- Podprogram
  - bez parametrov
  - RCX dovoˇuje max 8 podprogramov v 1 programe
  - podprogram nemôže byť volaný z iného podprogramu
  - môže byť volaný z rôznych úloh, ale POZOR na súčasný beh dvoch totožných podprogramov
  - `sub zatoc()`
  - `{ … }`

  - `task main()`
  - `{ … zatoc(); … zatoc(); … }`

  - ⇒ vyplatí sa iba ak potrebujeme skrátiť kód; príklad `otáčaj.nqc`

Inline funkcie

- Nešetria miesto - vždy sú „rozbaľované“
- neobmedzený počet

```
void otacaj()  //robot sa otaca dokola
{
  OnRev(OUT_C); Wait(340);
  OnFwd(OUT_A+OUT_C);
}
```

- môžu mať parametre

```
void otacaj(int t            //robot sa otaca dokola t
                                //stotin sekundy
{
  OnRev(OUT_C); Wait(t);
  OnFwd(OUT_A+OUT_C);
}
```

Makrá

- Aspoň tak silné ako inline funkcie

```
#define turn_around OnRev(OUT_C);Wait(340);OnFwd(OUT_A+OUT_C);
```

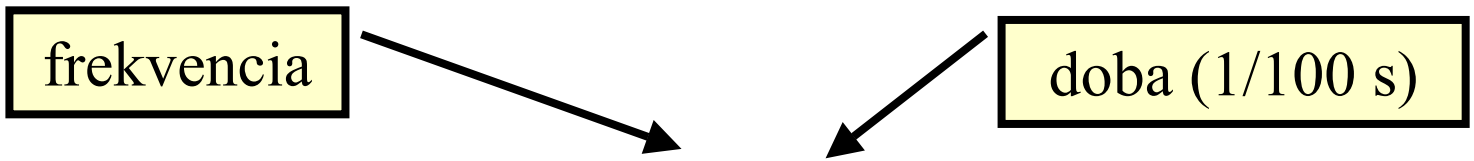
- Aspoň môžu mať tiež parametre
  -
```
#define zatoc_doprava(s,t)  SetPower(OUT_A+OUT_C,s);\
                            OnFwd(OUT_A);OnRev(OUT_C);Wait(t);
```

Melódie

- Vhodné na signalizáciu stavu
- zabudované melódie
  - 0 klik
  - 1 dvojité pípnutie
  - 2 klesajúce tóny
  - 3 stúpajúce tóny
  - 4 bzukot - chyba
  - 5 rýchle stúpajúce tóny

- `PlaySound("zvuk"); Wait(100);`
  - `Wait`, aby melódia dohrala
  - `"zvuk"` musí byť koštanta

Zvuky




```
playtime(262,40);
```

```
task main()
{
  PlayTone(262,40);  Wait(50);
  PlayTone(294,40);  Wait(50);
  PlayTone(330,40);  Wait(50);
  PlayTone(294,40);  Wait(50);
  PlayTone(262,160); Wait(200);
}
```

- výhodné dať do samostanej úlohy

Zmena rýchlosti

- Rozdiel medzi rýchlosťou 2 a 7 je malý
- lepšie je zapínať a vypínať motor v krátkych intervaloch
- vhodné do samostatnej úlohy

```
int speed, _speed;
task run_motor()
{
  while (true)
  {
    _speed = speed;
    if (_speed > 0) {OnFwd(OUT_A+OUT_C);}
    if (_speed < 0) {OnRev(OUT_A+OUT_C); _speed = -_speed;}
    Wait(_speed);
    Off(OUT_A+OUT_C);
  }
}
```

Senzory podrobnejšie

- `SetSensor(SENSOR_1,SENSOR_LIGHT)` nastaví typ a režim senzora
- `SetSensorType(„typ“)` nastaví iba typ senzora; argument:

  `SENSOR_TYPE_TOUCH, SENSOR_TYPE_LIGHT, SENSOR_TYPE_ROTATION`
- `SetSensorMode(„režim“)` nastaví režim senzora; argument:
  - `SENSOR_MODE_RAW` hodnota 0-1023 závislá na senzore
    - dotykový: stlačený ~50, uvolnený ~1023, ale i hodnoty medzi pri čiastočnom stlačení
    - svetelný: ~300 svetlo, ~800 tma
  - `SENSOR_MODE_BOOL` 0 ak je RAW hodnota vyššia než 550, inak 1
    - implicitný režim pre dotykový senzor
  - `SENSOR_MODE_PERCENT` do 400 dáva 100 ďalej klesá k nule
    - implicitný pre svetelný senzor

Senzory podrobnejšie

- `SetSensorMode(„režim“)`
  - `SENSOR_MODE_EDGE` počíta „výrazné“ zmeny vstupu nahor i nadol
  - `SENSOR_MODE_PULSE` počíta „výrazné“ zmeny vstupu nahor
    - napr. Počet stlačení, uvoľnení dotykového senzora
    - čítač sa dá vynulovať volaním `ClearSensor(„senzor“)`.
  - `SENSOR_MODE_ROTATION` iba pre polohové čidlo
    - tiež sa dá nulovať
    - dá sa použiť na sledovanie ujdenej dráhy, zaručenie rovnej dráhy (`priamy_pohyb.nqc`)

Triky so senzormi

- 2 senzory na 1 vstupe
  - oba dotykové v režime BOOL - jeden vpredu a jeden vzadu hodnoyu 1 dostanem pri stlačení jedného z nich, ale viem, či som išiel dopredu alebo dozadu;
  - oba dotykové v režime RAW - dá sa rozlíšiť i stlačenie oboch naraz - hodnota menšia než 30
  - dotykový+svetelný - typ nastaviť na LIGHT, režim na RAW, hodnota menšia než 100 - bol stlačený dotykový senzor, inak dostávame hodnotu svetelného senzoru
- infračervená veža jako radar

Paralelné úlohy

- problém
  - Môžu sa pokúsiť ovládať tie isté výstupy naraz - výsledné chovanie je nezmyselné
  - keď úlohu zastavíme, tak jej opätovné spustenie ju spustí znova od začiatku
- riešenie - semafór
  - príklad `Semafór.nqc`

Čo sme vynechali

- Udalosti - events
- komunikáciu medzi RCX kockami
- voľbu vedúceho
- časovače
- prácu s displejom
- zaznamenávanie dát (datalog) a predávanie týchto dát do PC

- **lepšie možnosti programovania brickOs**
  - http://brickos.sourceforge.net/index.html