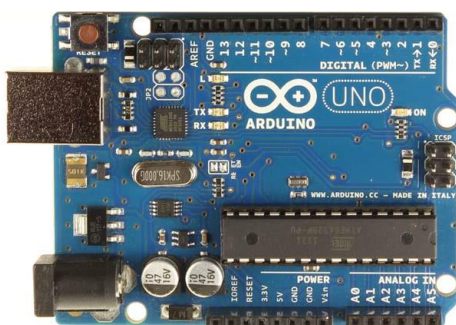


1 Programování Arduino – seznámení

Projekt Arduino (<http://arduino.cc/>) představuje open-source platformu pro vývoj jednoduchých aplikací postavených na osmi bitových mikrokontrolérech řady ATmega a počínaje deskou Arduino Due i na dvaatřicetibitových mikrokontrolérech ATSAM3X s jádrem typu ARM. Pro tvorbu aplikací poskytuje Arduino vývojové prostředí, které lze volně stáhnout na stránce <http://arduino.cc/en/Main/Software>. Vývojové prostředí je sice postaveno na jazyce Java avšak framework vychází z jazyka C++, který se pro konkrétní mikrokontroléry interně překládá kompilátorem WinAVR.

Dalším vývojovým prostředím, které umožňuje si otestovat programování arduina s možností si připojit virtuální Arduino a virtuální periférie je aplikace UnoArduSim, která lze volně stáhnout na stránce <https://www.sites.google.com/site/unoardusim/services>.

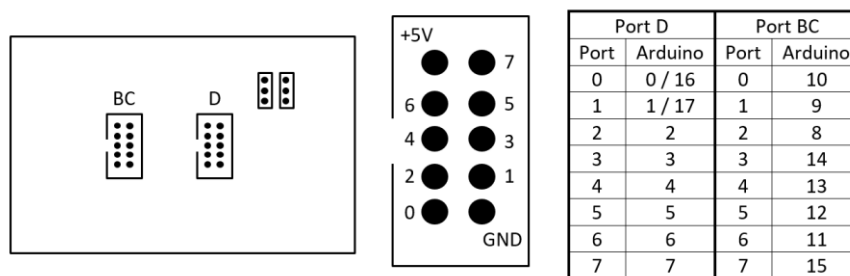
Se snahou o maximální zjednodušení vývoje poskytuje Arduino několik vývojových desek, které se navzájem liší hlavně použitým mikrokontrolérem a počtem ovládaných pinů (vstupů/výstupů). Desky si můžete podle dokumentace vyrobit sami nebo si je relativně levně koupit v mnoha obchodech. Ve výuce bude použita deska Arduino UNO, která je řízena mikrokontrolérem ATmega328p.



Obr. 1 Arduino UNO

Tato deska obsahuje 13 digitálních pinů a 5 takzvaně analogových pinů. Analogové piny však lze využívat i jako digitální, jen je navíc možno na ně připojit analogový signál a pomocí integrovaných AD převodníků zjišťovat aktuální hodnotu napětí.

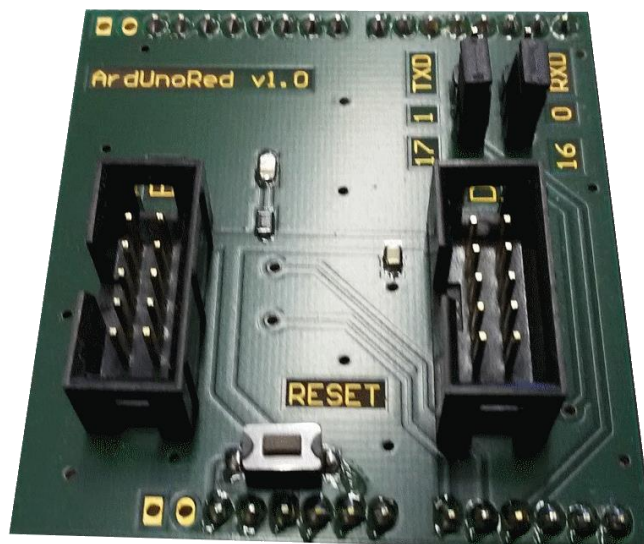
Pro výuku je připraveno několik přípravků, které lze k desce Arduino UNO snadno připojit prostřednictvím propojovací desky obsahující dva deseti-pinové konektory zapojené podle obrázku, proto je třeba u jednotlivých portů počítat příslušné porty, aby bylo možné připojit reálné periférie.



Obr. 2 Popis portů na redukci pro připojení periférií

1.1 Přípravky pro výuku

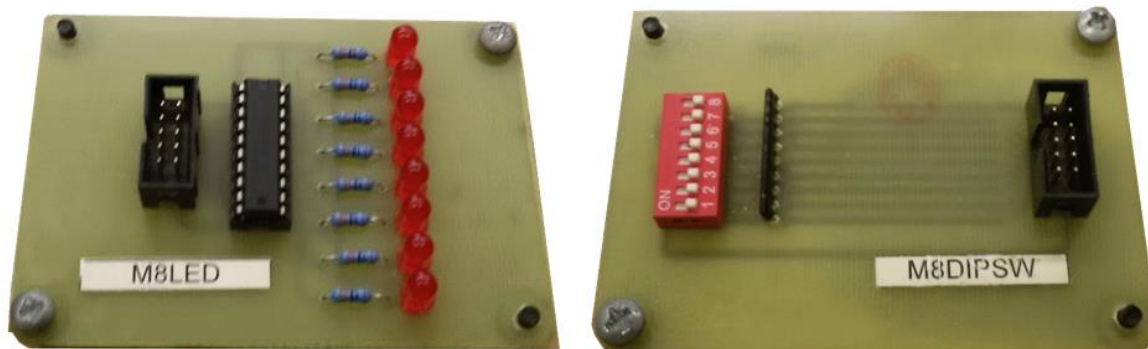
HW redukce pro připojení periférií. Proto je nutné dodržet rozmístění pinů na portech, aby aplikace vytvořená pro Arduino pracovala s perifériemi korektně. Proto budou i v ukázkových programech rozmístěny piny dle této redukce.



Obr. 3 Redukce pro připojení periférií




1.2 Základní periférie

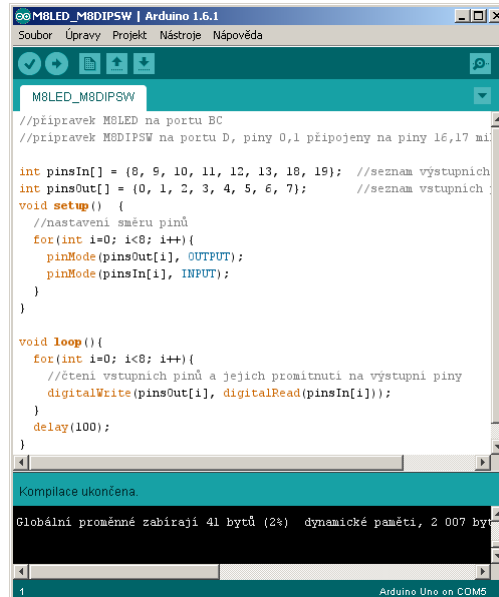
Pro základní výuku používáme nejčastěji periférii s 8xLED a periférii 8xDIP. Tyto periférie jsou na dalších obrázcích.



Obr. 4 Periférie 8xLED a 8xDIP

1.3 IDE Arduino

Vývojové prostředí je relativně strohé. Z jeho nástrojové lišty využijete hlavně tlačítko pro nahrání programu do mikrokontroléru , pro uložení zdrojového kódu  a později pro otevření konzolového okna s výpisem komunikace po sériové lince .



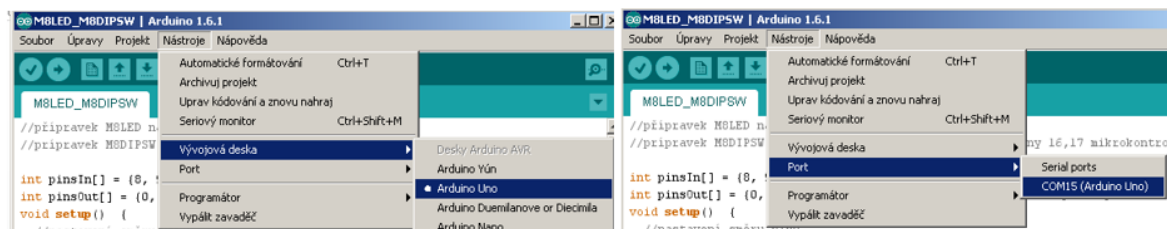
Obr. 5 IDE s programem pro 8xLED a 8xDIP

Každý program pro Arduino by měl obsahovat alespoň dvě základní procedury. První z nich je `void setup()` ta je volána pouze po resetu mikrokontroléru a slouží pro počáteční nastavení pinů a dalších komponent. Druhou hlavní procedurou je `void loop()` jež je volána stále dokola. Jakmile skončí, je znovu zavolána. Do této procedury se umísťuje výkonný kód.



Obr. 6 Základní struktura programu

Před nahráním programu do mikrokontroléru je potřeba vývojovému prostředí říci pro jakou vývojovou desku má zdrojový kód přeložit a dále přes jaký sériový port je deska k počítači připojena. Desku vyberete přes hlavní menu „Nástroje“ -> „Vývojová deska“ -> „Arduino Uno“. Sériový port může být na každém počítači jiný. V menu „Nástroje“ -> „Port“ je vidět seznam aktuálně dostupných COM portů, zde vyberte ten, který se nově objeví po připojení desky Arduino UNO (obvykle ten s nejvyšším číslem).

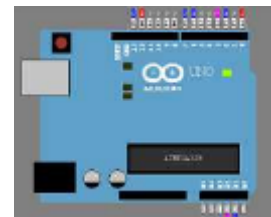


Obr. 7 nastavení IDE pro připojené Arduino

Aby byla tvorba programů ještě rychlejší, nemusíte začínat psaním od začátku. Prostředí disponuje řadou vzorových programů ukazujících práci s jednotlivými komponentami frameworku. Jejich seznam je vidět v menu „Soubor“ -> „Příklady“. Nejzákladnějším ukázkovým programem je „Blink“, který najdete v „Soubor“ -> „Příklady“ -> „01.Basics“ -> „Blink“. Ten bliká s LED diodou připojenou k pinu 13 umístěnou přímo na desce Arduina. Pokud máte však na Arduino nasazenou konektorovou desku pro přípravky není tato dioda příliš vidět. Můžete však připojit na port BC přípravek M8LED a bude Vám blikat dioda číslo 4.

1.4 Simulátor UnoArduSim

UnoArduSim je freeware nástroj reálného simulátoru, který je vyvinutý pro studenty a pro Arduino nadšence. Je navržen tak, aby vám umožnil experimentovat a snadno ladit, vaše Arduino programy bez nutnosti jakéhokoli skutečného hardwaru. Program UnoArduSim je určen pouze pro ArduinoUno a umožňuje vybrat ze sady virtuálních I/O zařízení a tyto virtuální zařízení nakonfigurovat pro konkrétní připojení k virtuálnímu ArduinoUno v LabBench panelu. Nemusíte se bát chyb, že špatně nějaké zařízení připojíte, z uvolněného spoje, nebo zničení přístroje špatnou konfigurací či zásahem do vašeho programu.



UnoArduSim poskytuje jednoduché chybové zprávy pro případné chyby a umožňuje ladění pomocí funkce Reset, Run, Run-To, Halt a flexibilní Stepping (krokování) v panelu kód (Code Pane), při současném zohlednění všech globálních a aktuálně aktivních lokálních proměnných, polí a objektů v panelu Proměnné (Variables Pane). Run-time kontrola překročení mezí je implementována, a jsou detekovány přetečení ATmega RAM (chybný programový řádek je zvýrazněn!). Jakékoliv elektrické konflikty s připojenými I/O zařízení jsou označeny a to včetně řádky, ve které k nim dojde.

Při otevření souboru INO nebo PDE, je načten soubor do Code panelu. Program je pak analyzován, a "sestaven" do spustitelné formy, která je pak připravena k simulování. Na rozdíl od Arduino.exe, není vytvořen žádný samostatný binární soubor a je aktivována chybová analýza a označeny řádky, které se nepodařilo zpracovat a hlášení chyb ve stavovém řádku v dolní části okna UnoArduSim. Edit/View okno lze použít pro otevření a úpravu zdrojového kódu syntaxe je zvýrazněna. Chyby vzniklé během simulace (například proházené přenosové rychlosti) jsou hlášeny na stavovém řádku, a přes pop-up MessageBox.

UnoArduSim V1.7 v podstatě podporuje programovací jazyk V1.6.6 Arduino, tak jak je zdokumentováno na internetové stránce jazykové reference arduino.cc., a s dodatky, jak je uvedeno v sekci Download s poznámky k vydání každé verze. Ačkoli UnoArduSim nepodporuje plnou C++ implementaci, která je základem GNU kompilátoru Arduino.exe, je pravděpodobné, že pouze nejpokročilejší programátoři by zjistili, že některé C/C++ funkce nejsou podporovány. Obecně platí, že jsou podporovány pouze ty, nejužitečnější C/C++ funkce.

1.5 Panely programu UnoArduSim a předvolby


1.5.1 Programové okno (Code Panel)



Základní okno „Code“ zobrazuje uživatelský program a zvýrazněný řádek sleduje právě vykonávaný kód. Je-li program úspěšně analyzován, je zvýrazněna první řádka v main (), a program je připraven ke spuštění. Všimněte si, že „int main ()“ je implicitně přidán do zdrojového kódu Arduina vlastním programem UnoArduSim, nezahrnujete ji jako součást do svého zdrojového kódu. Řízení aplikace je pod kontrolou menu Execute a přidružené liště ikon a klávesových zkratk.






Poté co je provedena jedna či více instrukcí je zvýrazněna řádka, která má být provedena. Pro ovládání krokování programu lze využít následující ikony.

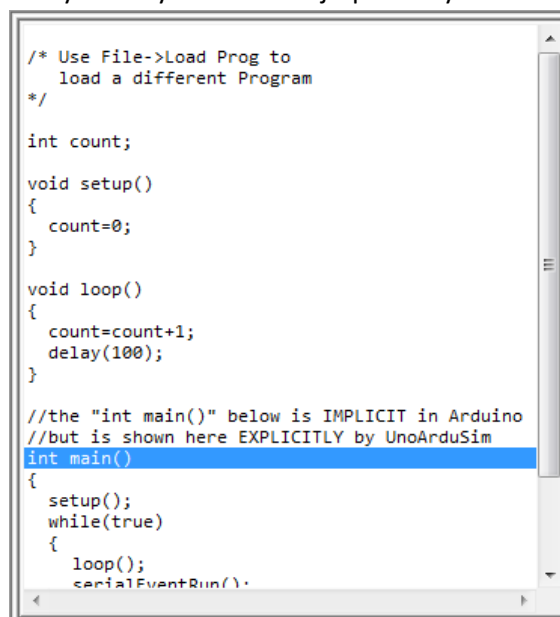


Stejně tak, když běžící program, se zastaví o zarážky (dočasné Run-K), provedení se zastaví a zvýrazní se na aktuální zarážce (a aplikace je připravena pro provedení dalších činností). Když je program spuštěn, UnoArduSim pravidelně upozorňuje označením aktuálního programového řádku, takže můžete vidět, že se nějaká aktivita se děje. Tato zvýraznění způsobí, že obsah okna se zdrojovým kódem přejdete do pevného zobrazení a zvýraznění aktivního řádku programu.

Jestliže klikneme na řádek v programovém okně je program korektně zastaven a příslušný řádek je zvýrazněn, pokud potřebujete spustit program až po zvýrazněnou řádku je to možné ikonou Run to.  Tato funkce umožňuje rychle a snadno dosáhnout postupného procházení v programu bez nutnosti krokovat řádek po řádku.

Pokud váš načtený program obsahuje (# include'd) knihovny souborů, tak můžete přecházet mezi nimi, pomocí položek menu File. Next (# include) File pro další a Previous File pro předchozí soubor. Nebo pomocí ikoněk modrých šipek  , .

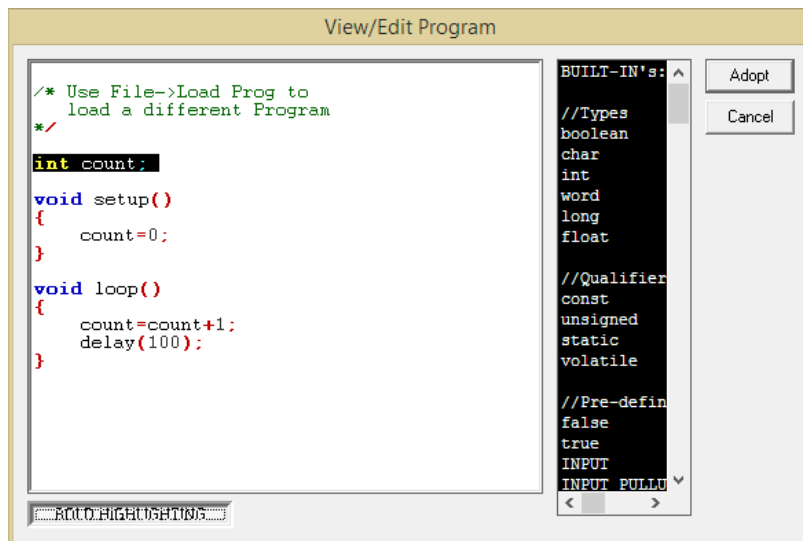
V nabídce Find (zkratky PgDn a PgUp nebo  a ) vám umožní rychle přecházet mezi jednotlivými funkcemi v podokně Code (ale musíte nejprve kliknout na řádek uvnitř panelu kód, aby jej zaměřit). Nebo si můžete skočit na zadáný text (po prvním použití Find> vyhledávání textu, nebo ) s příklady textu pro vyhledávání v nabídce, nebo jednoduše pomocí ikony  a  v panelu nástrojů.



Obr. 8 Programové okno

1.5.2 Editace / Zobrazení (Edit/View)

Dvojitým kliknutím na kteroukoli řádku v podokně programového kódu (nebo pomocí nabídky Soubor), okno View / Edit je následně otevřeno okno pro změny v programovém souboru, aktuálně vybraného řádku, který je zvýrazněn.

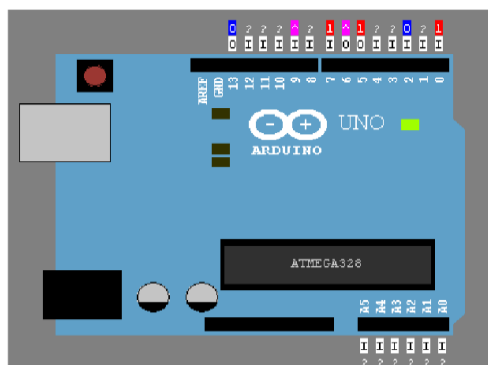


Obr. 9 Okno pro editaci a zobrazení zdrojového kódu

Toto okno umožňuje plnou editaci s dynamickým syntax zvýrazňováním (různé zvýraznění barev pro C++ klíčová slova, komentáře a podobně), volitelně tučné zvýraznění syntaxe a formátování automatické odrážky-level (za předpokladu že jste vybrali pomocí dialogového okna Config Preferences příslušnou volbu).

1.5.3 Panel ArduinoUno (Lab Bench Pane)

Lab Bench Panel zobrazuje Arduino Uno desku, která je obklopena řadou I/O zařízení, které můžete vybrat/přizpůsobit si a připojit na požadované Uno piny. Jedná se o znázornění desky ArduinoUno a jeho základních LED. Vložíte-li nový program do UnoArduSim, a je-li úspěšně analyzován tak podstoupí "simulované nahrání programu" do ArduinoUno, která napodobuje způsob, jakým skutečná ArduinoUno deska načítá program - uvidíte sériové RX a TX LED jak blikají (spolu s aktivitou na pinech 1 a 0, které jsou napevno propojeny se sériovou komunikací s hostitelským počítačem). Pak bezprostředně následuje bliknutí pinu 13 LED, která znamená, reset desky a čekání na začátku vašeho načteného programu.



Obr. 10 Základní panel simulátoru

2 Vzorové příklady

2.1 Běžící světlo

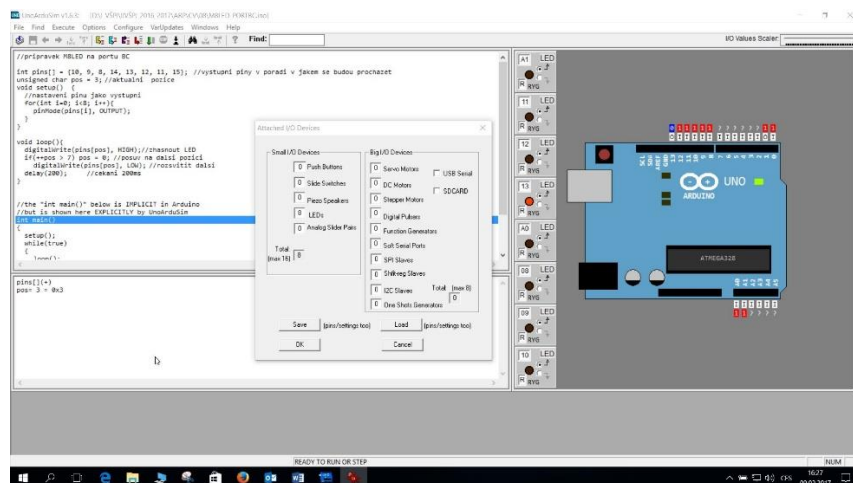
Připojte přípravek M8LED k portu BC a do mikrokontroléru nahrajte níže uvedený kód.

```
int pins[] = {10, 9, 8, 14, 13, 12, 11, 15}; //výstupní piny v pořadí v jakém budou procházeny
unsigned char pos = 0; //proměnná ve které je aktuální pozice
void setup() {
  //nastavení pinů jako výstupní
  for(int i=0; i<8; i++){
    pinMode(pins[i], OUTPUT);
  }
}
void loop() {
  digitalWrite(pins[pos], HIGH); //zhasnout aktuální
  if(++pos > 7) pos = 0; //posuv na další pozici
  digitalWrite(pins[pos], LOW); //rozsvítit další
  delay(200); //čekání 200ms
}
```

Kód začíná seznamem použitých pinů arduina uvedených v pořadí v jakém jsou připojeny ke konektoru BC. V proceduře `void setup()` se všechny použité piny nastaví jako výstupní. V proceduře `void loop()` se zhasne dioda na aktuální pozici zapsáním logické 1 na odpovídající pin. Poté se aktuální pozice posune o 1 dále a následně se rozsvítí dioda na nové pozici (zapsáním nuly). Nakonec se zavolá procedura `delay(int millis)` jejíž vykonání bude trvat zadaný počet milisekund (200), čímž se ve výsledku určuje rychlost běžícího světla. Zdrojový kód je umístěn v ukázkovém souboru M8LED_PORTBC.ino a je ke stažení z LMS MOODLE školy

2.1.1 Běžící světlo pomocí UnoArduSim

Řešení a testování základních úloh je nastaveno tak, aby bylo možné testovat aplikace bez nutnosti mít skutečné periférie a skutečný mikroprocesor. To umožní všem studentům testovat své aplikace i mimo laboratoř. Nejprve je nutné si aplikaci stáhnout z internetu a nainstalovat na počítač (školní počítače budou mít již aplikaci nainstalovanou). Po spuštění aplikace UnoArduSim, je třeba načíst zdrojový kód aplikace, to uděláte pomocí menu „File“ a volby „Load INO or PDE Prog.“ a načtete soubor M8LED_PORTBC.INO. Po načtení programu je ještě třeba nahrát HW konfiguraci. To můžete udělat buď ručně pomocí menu „Configure“ a volby „I/O devices“ nebo následným načtením přes tlačítko „Load“ je uložena v souboru M8LED_PORTBC_HW.txt. Pokud budete nastavovat hw konfiguraci ručně je třeba připomenout, že reálný modul M8LED pracuje tak, že LED dioda svítí, je-li na příslušném pinu logická nula.



Obr. 11 UnoArduSim Běžící světlo

2.2 Čtení bitů a výpis bitů

Připojte přípravek M8LED na port BC a přípravek M8DIPSW na port D. **Zkontrolujte, zda jsou na konektorové desce propojky pinů 0 a 1 nastaveny na piny 16 a 17 mikrokontroléru.** Do mikrokontroléru nahrajte níže uvedený kód. Lze jej načíst z LMS MOODLE soubor M8LED_M8DIPSW.ino

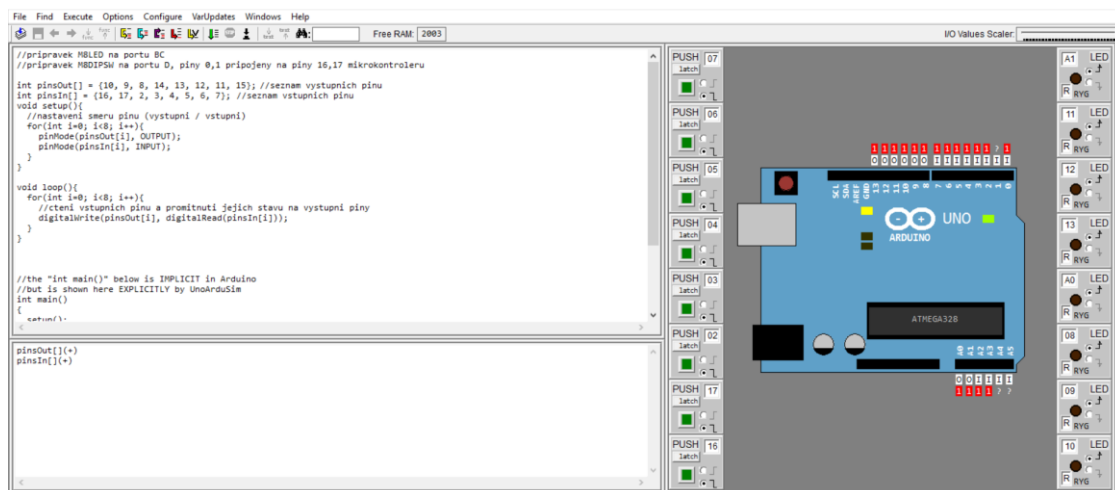
```
int pinsOut[] = {10, 9, 8, 14, 13, 12, 11, 15}; //seznam výstupních pinů
int pinsIn[] = {16, 17, 2, 3, 4, 5, 6, 7}; //seznam vstupních pinů
void setup(){
    //nastavení směru pinů (výstupní / vstupní)
    for(int i=0; i<8; i++){
        pinMode(pinsOut[i], OUTPUT);
        pinMode(pinsIn[i], INPUT);
    }
}

void loop(){
    for(int i=0; i<8; i++){
        //čtení vstupních pinů a promítnutí jejich stavu na výstupní piny
        digitalWrite(pinsOut[i], digitalRead(pinsIn[i]));
    }
}
```

Nejprve je nadefinován seznam výstupních pinů (`int pinsOut[]`) portu BC, na který jsou zapojeny LED diody a seznam vstupních pinů (`int pinsIn[]`) portu D, ke kterému jsou připojeny přepínače. V proceduře `void setup()` se nastaví módy nadefinovaných pinů procedurou `void pinMode(uint8_t pin, uint8_t mode)` na výstupní (OUTPUT) a vstupní (INPUT). V proceduře `loop` se vždy postupně čtou stavy vstupních pinů funkcí `int digitalRead(uint8_t pin)` a získaný stav se rovnou předá do procedury `void digitalWrite(uint8_t pin, uint8_t val)`, která nastaví hodnotu výstupního pinu. Pokud se tedy z přepínače načte logická 1, pak se odpovídající výstupní pin nastaví na log. 1 (+5V) a dioda zhasne (na přípravku M8LED mají diody obrácenou logiku – svítí při nule). Při načtení logické nuly se odpovídající pin nastaví na 0 a dioda se rozsvítí. Zdrojový kód je umístěn v ukázkovém souboru M8LED_M8DIPSW.ino.

2.2.1 Čtení bitů a výpis bitů pomocí UnoArduSim

Po spuštění aplikace UnoArduSim je třeba načíst zdrojový kód aplikace, to uděláte pomocí menu „File“ a volby „Load INO or PDE Prog..“ a načtete soubor M8LED_PORTBC.ino. Po načtení programu je ještě třeba nahrát HW konfiguraci. To můžete udělat buď ručně pomocí menu „Configure“ a volby „I/O devices“ nebo následným načtením přes tlačítko „Load“ je uložena v souboru M8LED_PORTBC_HW.txt.



Obr. 12 UnoArduSim Čtení bitů a výpis

3 Cvičení 1 pro Arduino UNO

Oba úkoly, lze odevzdat v jednom souboru „*.ino“ společně s HW konfigurací (např. `ukol_12_bilek.ino` a `ukol_12_bilek_hw.txt`). Nezapomínejte na to, že LED dioda na reálném modelu bude svítit při logické nule a pořadí bitů je také dán dle HW redukce. V případě řešení pomocí UnoArduSim si můžete vhodně zvolit piny i jinak.

Odevzdávejte jednotlivé úkoly do samostatného adresáře. A do každého z adresářů nahrajte tři soubory. Zdrojový kód (*.ino) konfiguraci MCU (myArduPrefs.txt) a konfiguraci periférií (myIODevs.txt)

3.1 Úkol 1 – změna směru běžícího světla

Upravte kód pro běžící světlo na přípravku M8LED tak, aby se směr běhu měnil podle stavu pinu 7 (osmý) na přípravku M8DIPSW. (např. `ukol_1_bilek.ino` a `ukol_1_bilek_hw.txt`)

3.2 Úkol 2 – změna rychlosti běžícího světla

Podle stavu pinů 0 a 1 (první a druhý) na přípravku M8DIPSW (piny 16 a 17 mikrokontroléru) nastavujte zpoždění 50ms, 100ms, 500ms, 1000ms. (např. `ukol_2_bilek.ino` a `ukol_2_bilek_hw.txt`)