Ústav radioelektroniky
Vysoké učení technické v Brně

Zobrazovací zařízení, zpracování analogových signálů

Mikroprocesorová technika a embedded systémy

Přednáška 6

doc. Ing. Tomáš Frýza, Ph.D.

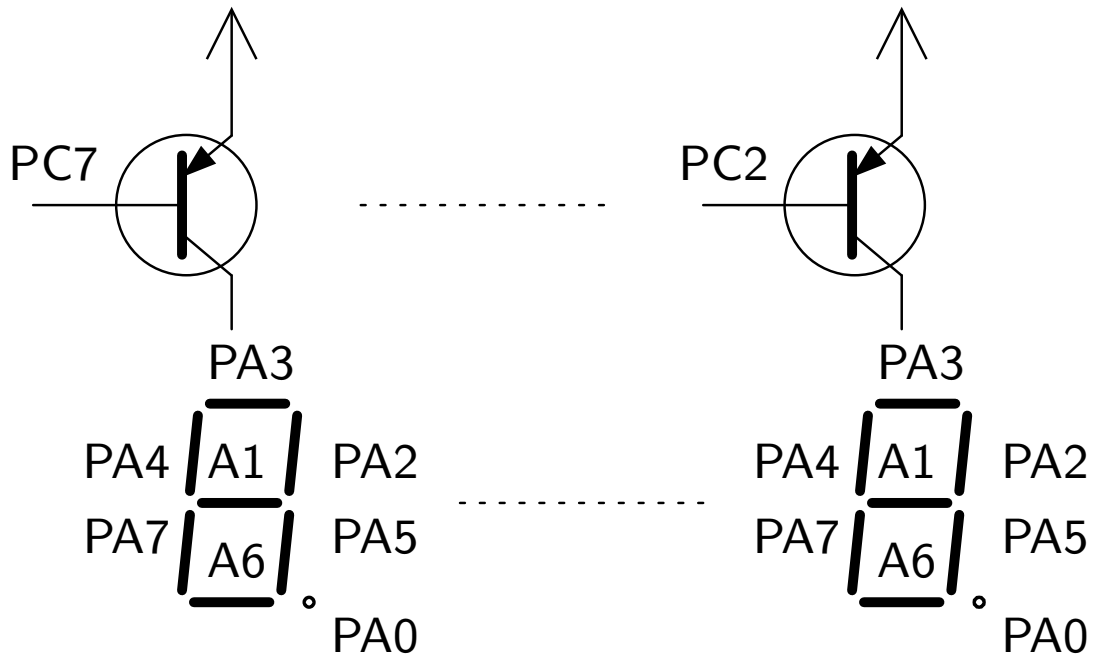
25. října 2011

Obsah přednášky

Obsah přednášky

Ovládání zobrazovacích zařízení – přímé propojení s MCU

- Přímé propojení: např. LED, 7segmentový displej.



Obrázek: Zapojení 7segmentového displeje v laboratoři Mikroprocesorové techniky; 6 symbolů.

Pozn.: Příklad použití 7segmentového displeje

```
1  #include <avr\io.h>          // hlavičkový soubor pro mikrokontrolér
2  #include <avr\interrupt.h>   // hlavičkový soubor pro přerušení
3
4  // prototyp funkce zobrazující jeden symbol na požadované pozici
5  int showSeg( char, char ) ;
6
7  int main( void ){            // hlavní funkce
8      ...                      // kód hlavní funkce
9      sei() ;                  // globální povolení všech přerušení
10     while( 1 ) ;             // nekonečná smyčka
11
12     return( 1 ) ;            // hlavní funkce vrací hodnotu 1
13 }
14
15 ISR( ... ){                 // obsluha přerušení při přetečení časovače
16     ...                      // kód obsluhy přerušení
17     // zobrazení jednoho symbolu "symbol" na pozici "position"
18     showSeg( symbol, position ) ;
19 }
20
21 int showSeg( char symbol, char position ){
22     switch( symbol ){
23         case 0:              // zobraz symbol "0"
24         ...                  // doplňte kód celé funkce
25     }
26     return( 0 ) ;            // návratová hodnota funkce
27 }
```

Znakové LCD displeje

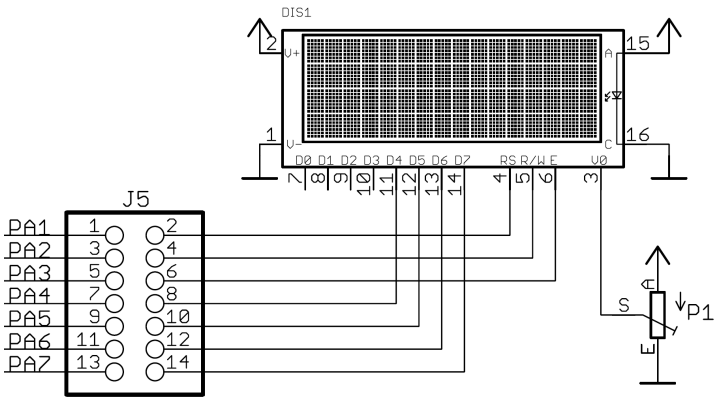
- Řízení komunikace s displejem (znakový, grafický).
- Zobrazitelná plocha obsahuje běžně od 8×1 do 40×4 znaků, bez nebo s podsvícením (nejčastěji žluto-zelené) – proudový odběr.
- Většina LCD displejů obsahuje řídicí obvod HD44780 firmy Hitachi (příp. jeho derivát); prostřednictvím tohoto obvodu je možné komunikovat s displejem. Tento "řadič" obsahuje jak znakovou sadu i instrukce pro ovládání.
- Komunikace probíhá pomocí 8 datových signálů ($DB0 - DB7$) a je řízena 3 řídicími signály (RS, $R/\overline{W}$ a E):
  - RS – identifikuje, zda se přenáší instrukce nebo zobrazitelná data,
  - $R/\overline{W}$ – volba operace čtení nebo zápisu (z pohledu MCU),
  - E – spouštěcí ("vzorkovací") signál.

(a) (b) (c)

Obrázek: Různé provedení znakových LCD displejů.

Datová komunikace mezi LCD a MCU

- Existují dva základní způsoby datové komunikace mezi LCD a MCU, využívající plný (8bitová komunikace) nebo redukovaný (4bitová komunikace) počet datových signálů.
- 8bitová komunikace vyžaduje 8 + 3 I/O pinů mikrokontrolérů.
- 4bitová komunikace: 4 + 3 I/O pinů:
  - u LCD se využívá horní polovina datových pinů DB7 až DB4,
  - rozdělení datového slova do dvou niblů v časovém multiplexu, tj. dva pulzy řídicího signálu *E*.



Obrázek: Zapojení LCD displeje na vývojové desce ATmega16.

Tabulka: Popis pinů LCD displeje MC1604 (16×4 znaků).

Č. pinu	Symbol	Popis
1	V−	Napájecí napětí (GND).
2	V+	Napájecí napětí (+).
3	V0	Kontrast znaků.
4	RS	Výběr instrukce/data.
5	R/W	Výběr čtení/zápis.
6	E	Signál Enable.
7-14	DB0–DB7	Datové piny.
15	A	Podsvícení – anoda.
16	C	Podsvícení – katoda.

Řízení komunikace mezi LCD a MCU

- Význam hodnot řídicích signálů:
  - $RS = 0$: přenáší se instrukce (např. pro smazání obsahu displeje), $RS = 1$: data (např. text k zobrazení).
  - $R/\overline{W} = 0$: zápis dat/instrukcí do LCD, $R/\overline{W} = 1$: čtení z LCD.
  - sestupná hrana kladného pulzu signálu E startuje komunikaci. Data na sběrnici musí být ustálena již před nástupnou hranou. Minimální doba trvání impulzu?
  - Možnost úspory dalšího pinu – $R/\overline{W} = \text{GND}$.

Příklad

Nakreslete časové průběhy všech signálů při zápisu instrukce 0b1111 0000 do LCD pomocí (a) 8bitové a (b) 4bitové komunikace.

- Znaky jsou zobrazovány jako matice 5×8 (většina znaků je menších); jsou v LCD uloženy ve vnitřní paměti RAM.
- Znaky adresované 16–31 (0x10–0x1F) a 128–159 (0x80–0x9F) nelze zobrazit (odpovídají řídicím znakům z ASCII tabulky).
- Data, která chceme zobrazit, jsou v LCD uložena v paměti s označením DDRAM (Display Data RAM).
- Je možné definovat až 8 uživatelských znaků (adresa 0–7). Tento segment je označován jako CGRAM (Character Generator RAM).

Instrukční sada řídicího obvodu HD44780

Tabulka: Vybrané příkazy HD44780 pro komunikaci s LCD displejem.

RS	$R/\overline{W}$	$DB7 : DB0$	**Popis instrukce**
0	0	`0000 0001`	Smazání displeje.
0	0	`0000 001x`	Návrat kurzoru na pozici (0,0).
0	0	`0000 01IS`	Nastavení posuvu kurzoru. `I`: inkrementace pozice kurzoru. `S`: posuv displeje.
0	0	`0000 1DCB`	Zapnutí displeje/kurzoru. `D`: zapnutí displeje. `C`: zobrazení kurzoru. `B`: blikání kurzoru.
0	0	`001L NFxx`	Nastavení rozlišení. `L=1(0)`: nastavení 8(4)bitové komunikace. `N=1(0)`: dva(jeden) řádky displeje. `F=1(0)`: velikost fontu $5\times10(5\times7)$.
0	0	`01 cgram`	Nastavení adresy `cgram` v CGRAM segmentu.
0	0	`1 ddram`	Nastavení adresy v DDRAM segmentu.
1	0	`data`	Zápis dat do paměti CGRAM. Zobrazení znaku z adresy `data`.
1	0	`data`	Zápis dat do paměti CGRAM nebo DDRAM.
1	1	`data`	Čtení dat z paměti CGRAM. Čtení dat z adresy `data`.
1	1	`data`	Čtení dat z paměti CGRAM nebo DDRAM.

$RS = \mathbf{0}$ instrukce

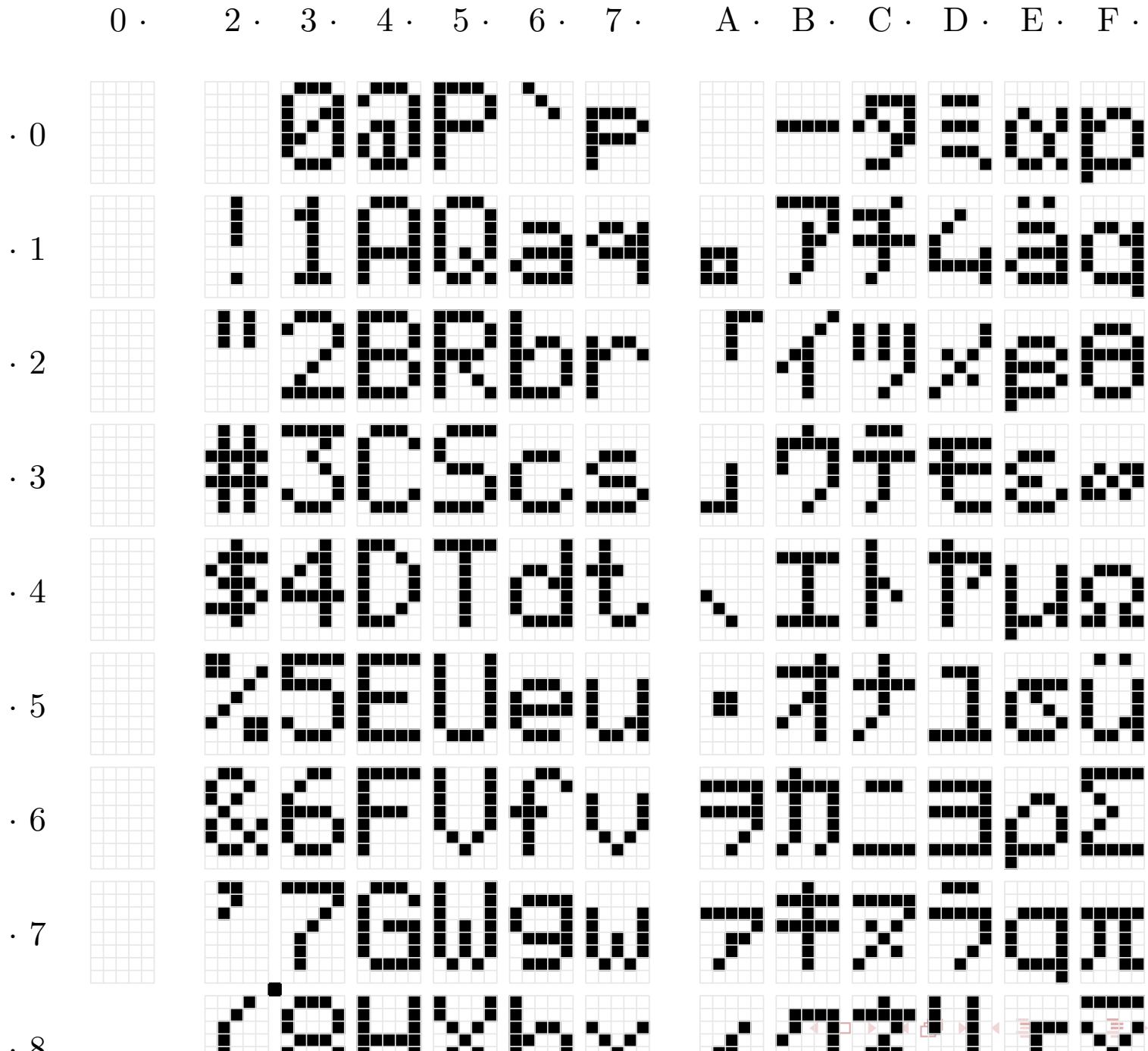
$RS = \mathbf{1}$ data

Inicializace LCD displeje

- Inicializace komunikace s displejem (základní nastavení řadiče HD44780).
- Provést jednou – začátek aplikace.
- Pozor na časování – jednotlivé příkazy vyžadují minimální časové prodlevy. Zpoždění v knihovně realizováno též prostřednictvím funkce `lcd_clrscr()`.

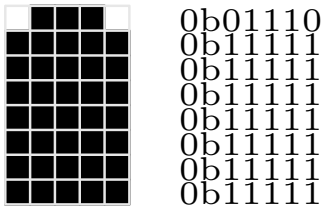
```
1  void lcd_init( void ){
2
3      DDRA = 0xFE ;                  // viz připojení LCD v laboratoři
4      _delay_ms( 16 ) ;              // delay
5      lcd_command( 0b00100000 ) ;    // 4bitová komunikace; viz vybrané příkazy
6      lcd_command( 0b00101000 ) ;    // více řádků displeje
7      lcd_command( 0b00000100 ) ;    // nastavení posuvu kurzoru
8      lcd_command( 0b00001100 ) ;    // zapnutí displeje
9  }
```

Znaková sada LCD displeje

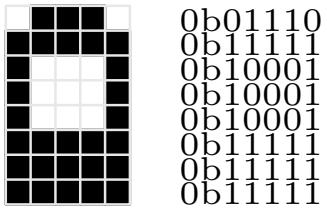


Znaková sada LCD displeje

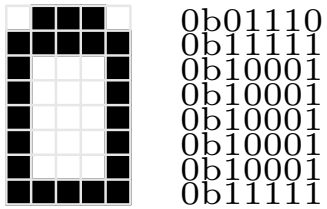
- Znaky se definují v matici 5×8 bodů. Zobrazený pixel bude v této matici obsahovat hodnotu "1", body které mají být skryty, obsahují "0". Jeden znak je tedy reprezentován osmicí slov (jedno slovo na řádek).
- Nový znak má smysl uložit do paměti CGRAM na samém začátku aplikace.



(a) 100 %



(b) 40 %



(c) 0 %

Obrázek: Nově definované znaky pro LCD displej – indikátor nabití baterie.

```
1     lcd_init() ;                    // inicializace LCD displeje – viz knihovna
2     lcd_clrscr() ;                  // smazání obsahu LCD
3
4     lcd_command( 0b01000000 ) ;     // příkaz: Nastavení adresy 0 v CGRAM
5     lcd_data( 0b01110 ) ;           // ulož první řádek symbolu od adresy 0
6     lcd_data( 0b11111 ) ;
7     lcd_data( 0b11111 ) ;
8     lcd_data( 0b11111 ) ;
9     lcd_data( 0b11111 ) ;
10    lcd_data( 0b11111 ) ;
11    lcd_data( 0b11111 ) ;
12    lcd_data( 0b11111 ) ;           // ulož poslední řádek nového symbolu
13    lcd_clrscr() ;                  // smazání obsahu LCD (nutné)
```

Knihovna v jazyce C pro práci s LCD

- Peter Fleury: `http://homepage.hispeed.ch/peterfleury/` (Pozn.: Datové piny MUSÍ být namapovány na dolní polovině portu.)

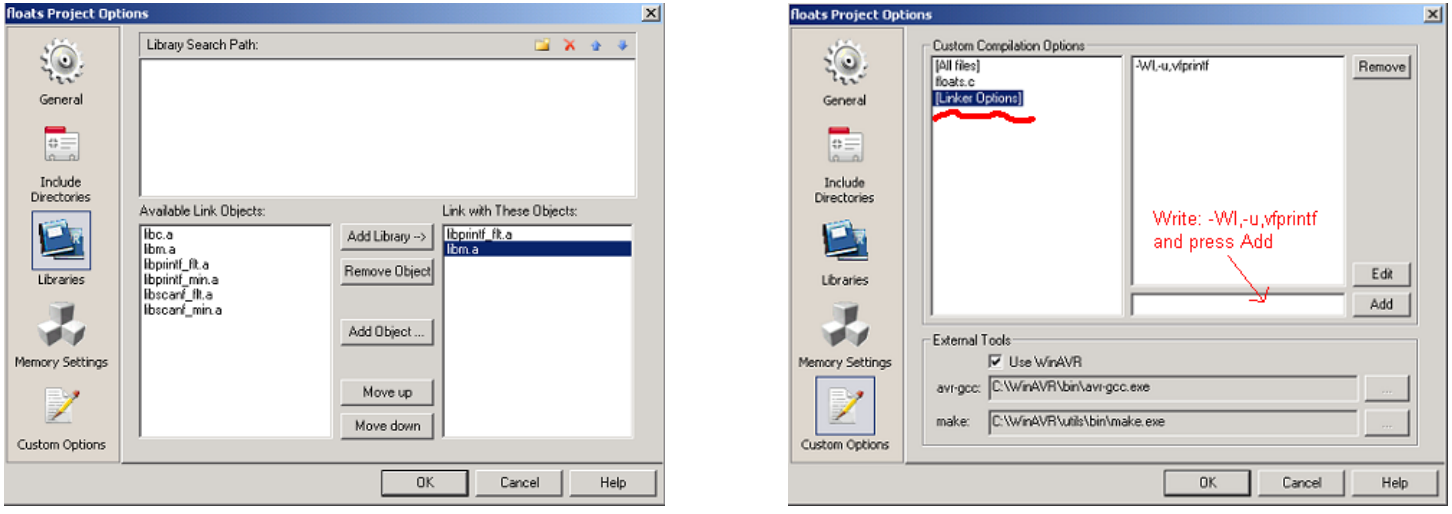
Tabulka: Soupis vybraných funkcí z knihovny pro LCD.

Název funkce	Popis funkce
`lcd_init() ;`	Inicializace LCD displeje.
`lcd_clrscr() ;`	Smazání obsahu LCD displeje.
`lcd_gotoxy( 0,2 ) ;`	Přesun kurzoru na pozici (0,2), tj. 1. sloupec, 3. řádek.
`lcd_putc( 'A' ) ;`	Zobrazení znaku "A".
`lcd_puts( "BMPT") ;`	Zobrazení řetězce znaků "BMPT".
`lcd_command( 0x40 ) ;`	Zápis 1bytové instrukce do LCD displeje.
`lcd_data( 0x0A ) ;`	Zápis 1bytových dat do LCD displeje.

- LCD pro Arduino: `http://www.arduino.cc/en/Tutorial/LCDLibrary`.
- `http://extremeelectronics.co.in/avr-tutorials/using-lcd-module-with-avrs/`
- ...

Pozn.: Zobrazení proměnné typu `float` na LCD

- Nutné přidat knihovny `libprintf_flt` a `libm` do překladu aplikace.
- Správné nastavení linkování: `-Wl,-u,vfprintf`.



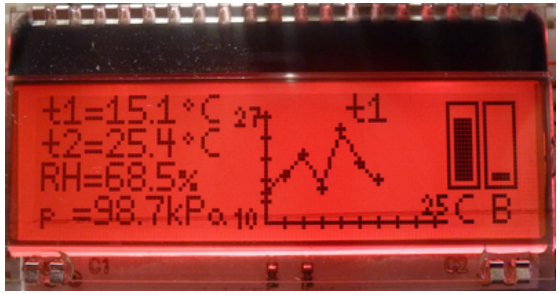
(a) (b)

Obrázek: Přidání knihoven do překladu v prostředí AVR Studio.

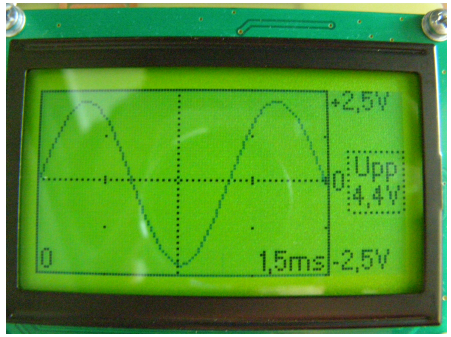
```
1  #include <stdio.h>          // standardní I/O knihovna (nutné pro sprintf)
2  #include "lcd_h.h"          // knihovna pro LCD
3  ...
4  float temp ;                // globální proměnná typu float
5  char buffer[16] ;           // glob. proměnná řetězce k zobrazení na displeji
6  ...
7      // naplnění řetězce buffer příslušným textem
8      sprintf( buffer, "Hodnota = %5.2f ", temp ) ;
9
10     // zobrazení řetězce na displeji
11     lcd_puts( buffer ) ;
12     ...
```

Pozn.: Grafické displeje

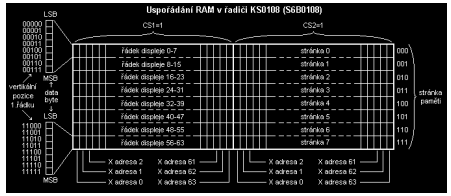
- Odlišné řídicí obvody:
  - Např. DOGM132X-5 od společnosti ELECTRONIC ASSEMBY – monochromatický grafický displej s rozlišením 132×32 pixelů (řadič ST7565R).



- Grafický displej ATM12864 s rozlišením 128×64 pixelů (řadič KS0108):
  - Absence znakové sady. Rozdělení obrazu na dvě části (64×64) – každá vlastní řadič. Signály paralelně, kromě *CS*1 a *CS*2 – levá, pravá část displeje.



(a)



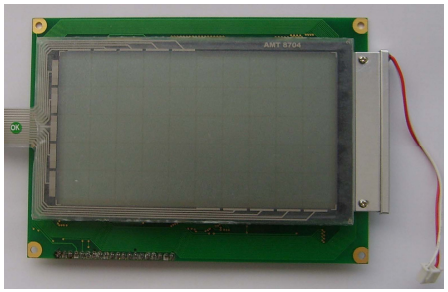
(b)

Obrázek: Grafický displej ATM12864.

Pozn.: Dotykový panel (touch screen)

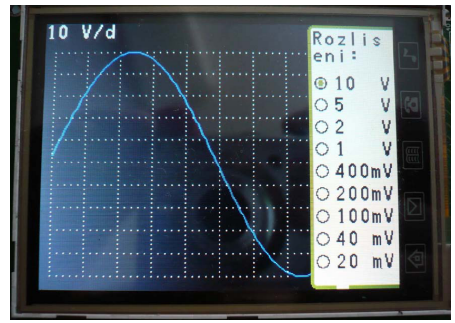
- Dělení podle druhu získání informace o poloze
  - Analogové: je nutné použít A/D převodník, příp. speciální obvod. Např. ADS7864E (12-Bit, 6-Channel Simultaneous Sampling ADC), fy TI,
  - Digitální: skenování řádků a sloupců – identické s maticovou klávesnicí,

Ukázka: Grafický displej PG240128WRF-ATA-H-Y7 (řadič T6963c) s 10×5 dotykovými poli @ 15 I/O pinů (digitální informace).



(a)

(b)



(c)

Obrázek: Aplikace s grafickým displejem: (a, b) PG240128WRF-ATA-H-Y7, (c) 3,2" (`http://www.ebay.com/itm/3-2-TFT-LCD-Module-Display-Touch-Panel-PCB-adapter-/200475566068?pt=LH_DefaultDomain_0&hash=item2ead465ff4`).

- Ukázky aplikací s grafickými displeji např.:
  - Samostatné projekty `http://www.urel.feec.vutbr.cz/~fryza/`
  - DP, Kulig
  - DP, Marcoň
  - Stránky předmětu Mikropočítače pro přístrojové aplikace, dr. Fedra `http://www.urel.feec.vutbr.cz/MIA/`

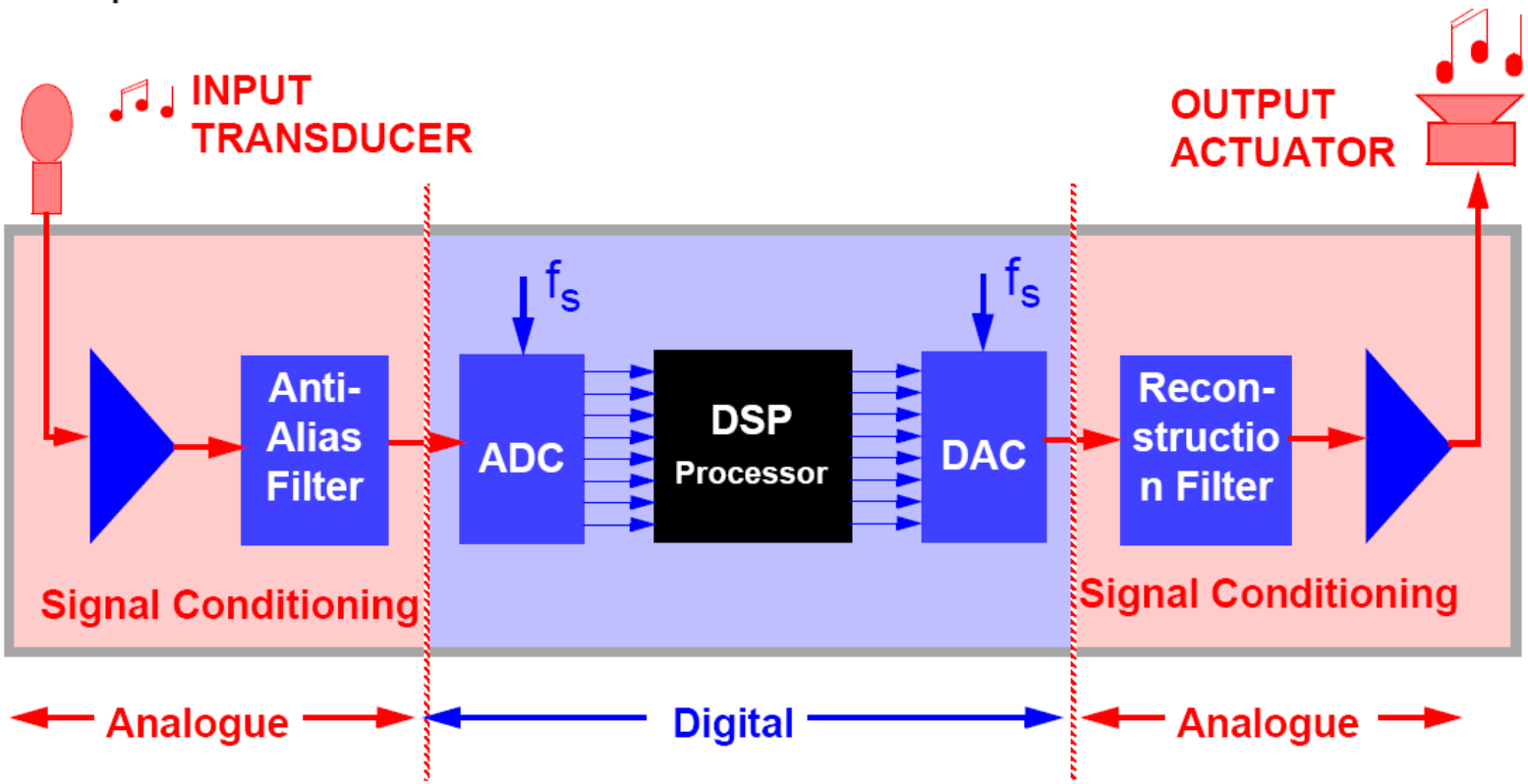
Obsah přednášky

Připomenutí A/D převodu

- Převzato z materiálů semináře DSP Primer firmy Xilinx, 22. května 2011, Praha.

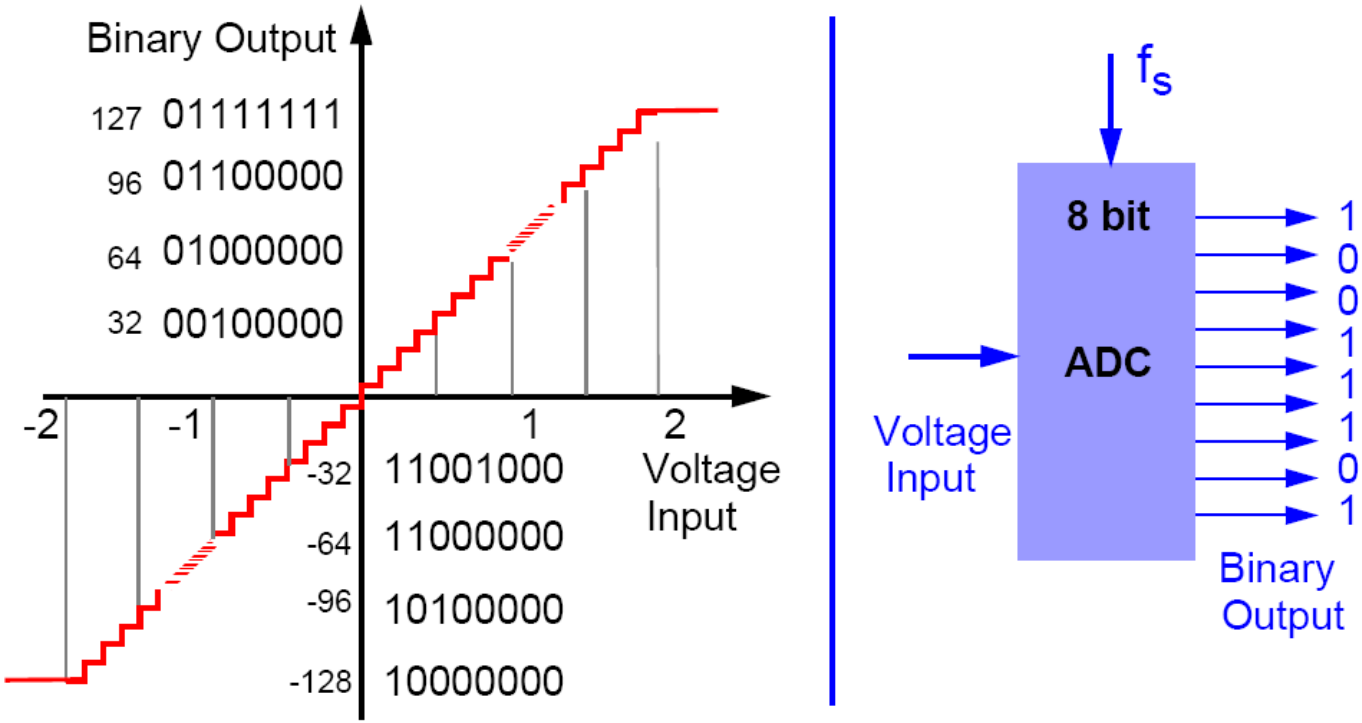
A Generic Input/Output DSP System

- A single input, single output DSP system has the following components:



Analogue to Digital Converter (ADC)

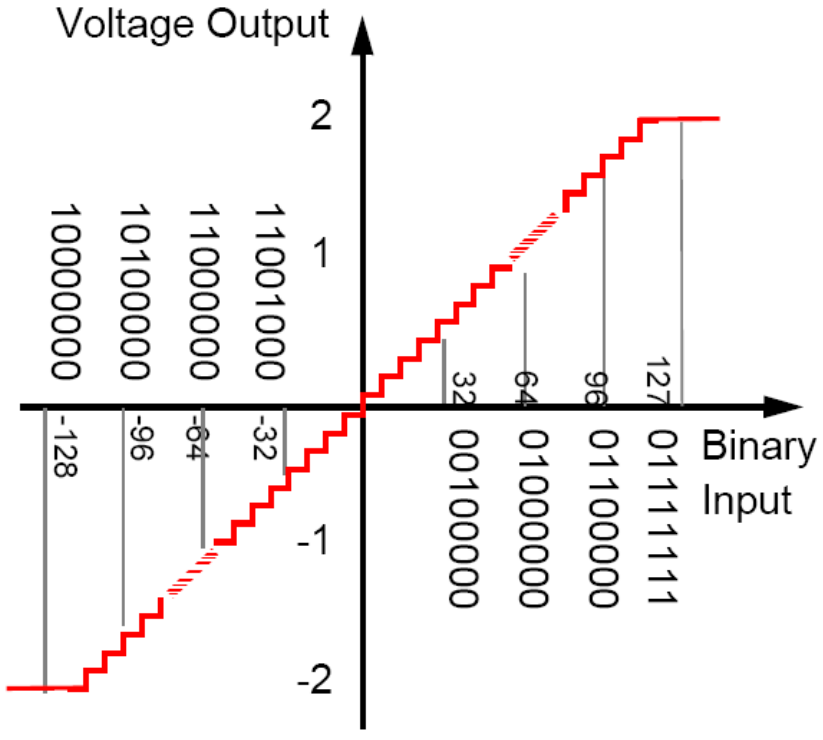
•An ADC is a device that can convert a voltage to a binary number, according to its specific input-output characteristic.

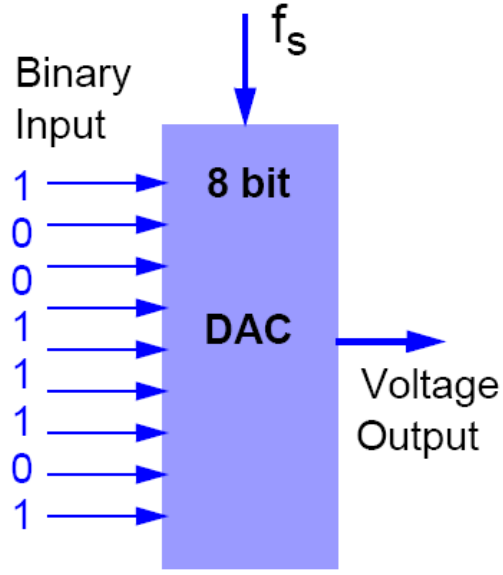


•The number of digital samples converted per second is defined by the sampling rate of the converter, f_s Hz.

Digital to Analogue Converter (DAC)

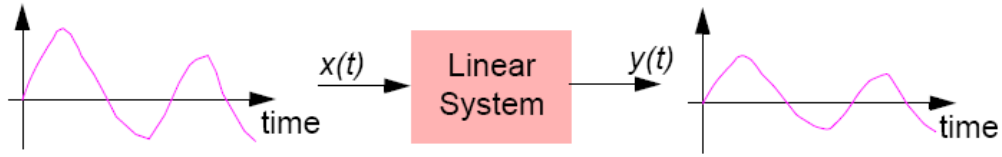
- A DAC is a device that can convert binary numbers to voltages, according to its specific input-output characteristic.





Notes:

A system is said to be linear if the output can formed as the convolution of the input and system transfer function



In general for a *linear system* $y(t) = f(x(t))$, if,

$$y_1(t) = f[x_1(t)]$$
$$y_2(t) = f[x_2(t)]$$

then by superposition: $y_1(t) + y_2(t) = f[x_1(t) + x_2(t)]$

One of the simplest ways to test the linearity of a system is to input a pure tone sine wave, and if, at all frequencies, the output is only a pure tone at the signal input frequency (i.e. no harmonics are produced), then the system is truely linear.

Amplification is often presented as a logarithmic measure of the power amplification ratio (P_{out}/P_{in}) given the large linear dymanic range. Recall that $P \propto V^2$, then:

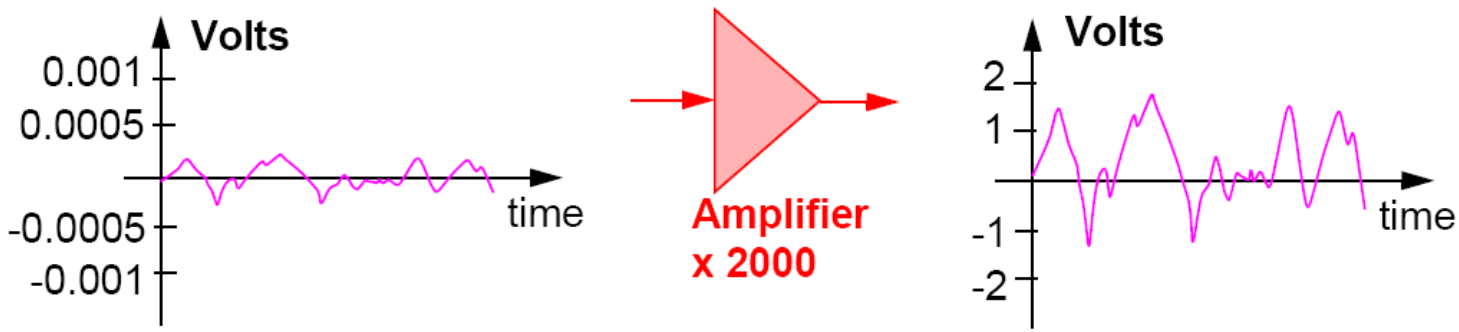
$$A_{dB} = 10\log_{10}(P_{out}/P_{in}) = 10\log_{10}(V_{out}/V_{in})^2 = 20\log_{10}(V_{out}/V_{in}) \text{ decibels (dB)}$$

Therefore if an amplification $A = 1000$, then the power amplification is 60 dB. Similary an attenuation of a factor of 1000, (or a gain of 0.001) corresponds to -60dB of gain, or 60dB of attenuation! The placement of the -ve sign needs some care given the antonymity of the words gain and attenuation.

Signal Conditioning

- Note that prior to a signal being input to an ADC, an amplifier will be required to ensure that the full voltage range of the ADC is used - this is referred to as signal conditioning.

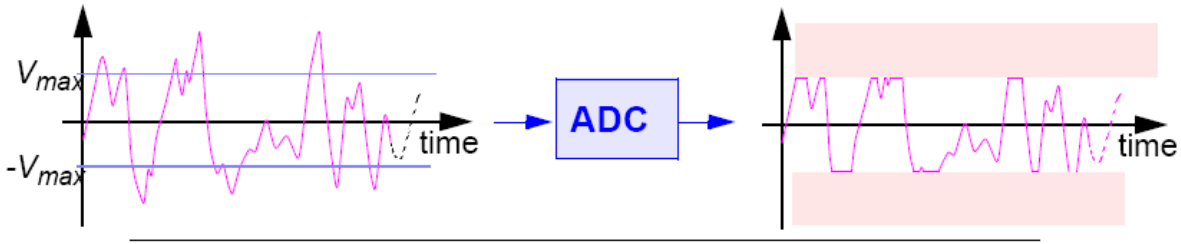
- For the above ADC with a maximum input and output of 2 volts we would require that the input signal to the ADC has a similar range:

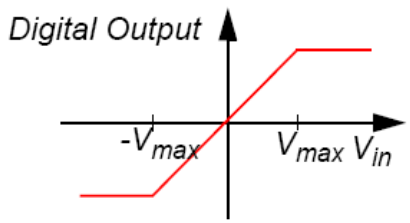


- Depending on the output actuator, an amplifier, or at least a buffer amplifier will be required.

Notes:

An analog signal with a magnitude larger than the upper and lower bounds V_{max} of an ADC chip, will be clipped. Any voltage above V_{max} will be clipped and the information lost. Clipping effects frequently occur in amplifiers when the amplification of the input signal results in a value greater than the power rail voltages.



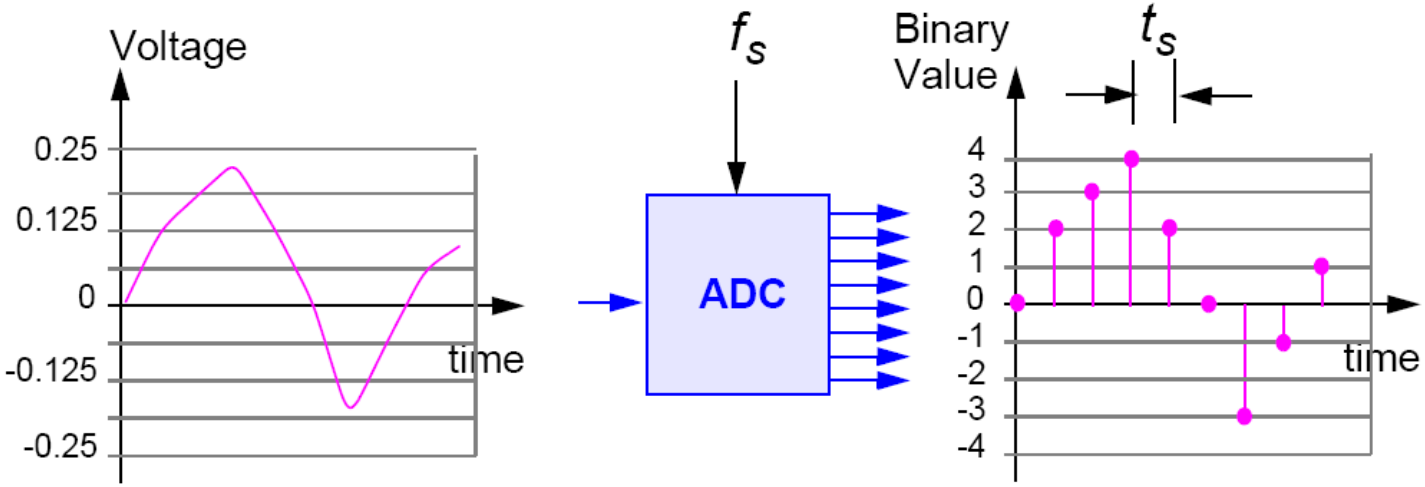


The DSP designer must ensure that for the particular application, clipping does not in general occur.

Sampling an Analogue Signal

- After signal conditioning the ADC can produce binary number equivalents of the input voltage.

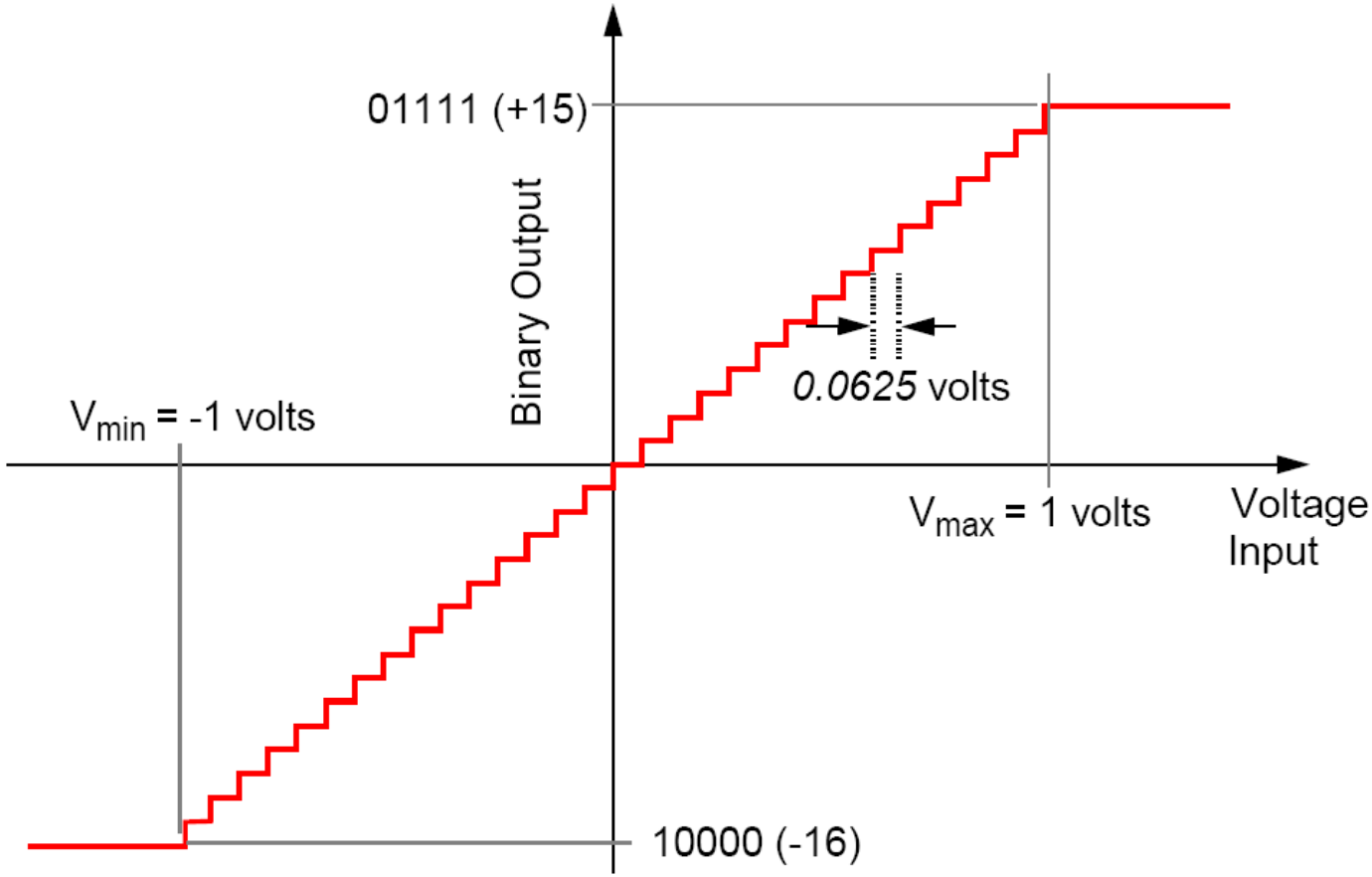
- If the ADC has finite precision due to a limited no. of discrete levels then there may be a "small" error associated with each sample.



- The quantisation step size is 0.0625 volts. If an 5 bit ADC is used, then the max/min voltage input is approx 0.0625 x 16 = 1 volt.

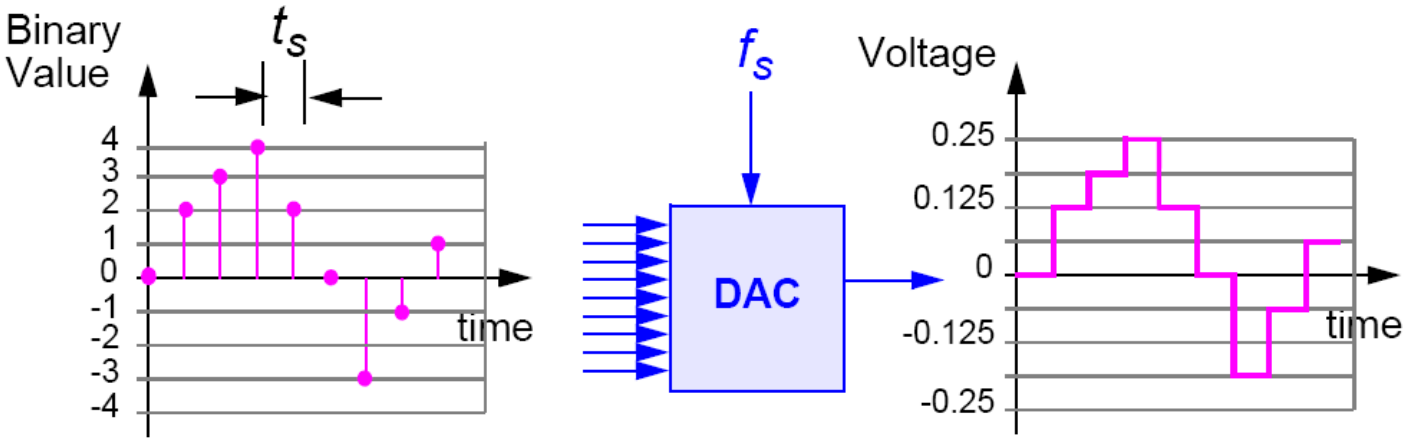
Notes:

For example purposes, we can assume our ADC or quantiser has 5 bits of resolution and maximum/minimum voltage swing of +1 and -1 volts. The input/output characteristic is shown below:



Reproducing an Analogue Signal

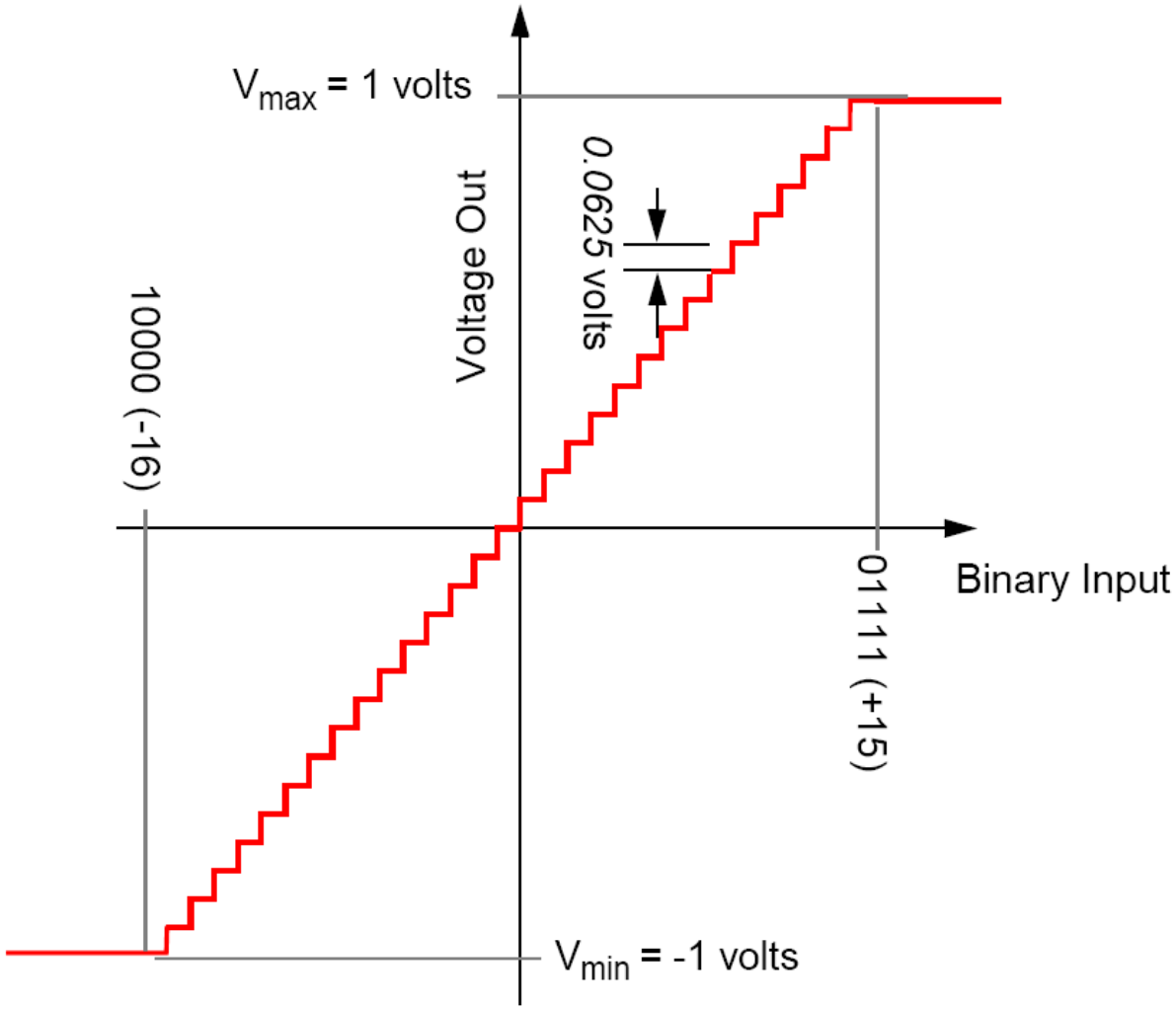
- Using a DAC at an appropriate sampling rate, we can reproduce an analogue signal:



- Note that the output is a little "*steppy*" caused by the zero order hold (step reconstruction);
  ....this artifact can however be removed with a reconstruction filter.

Notes:

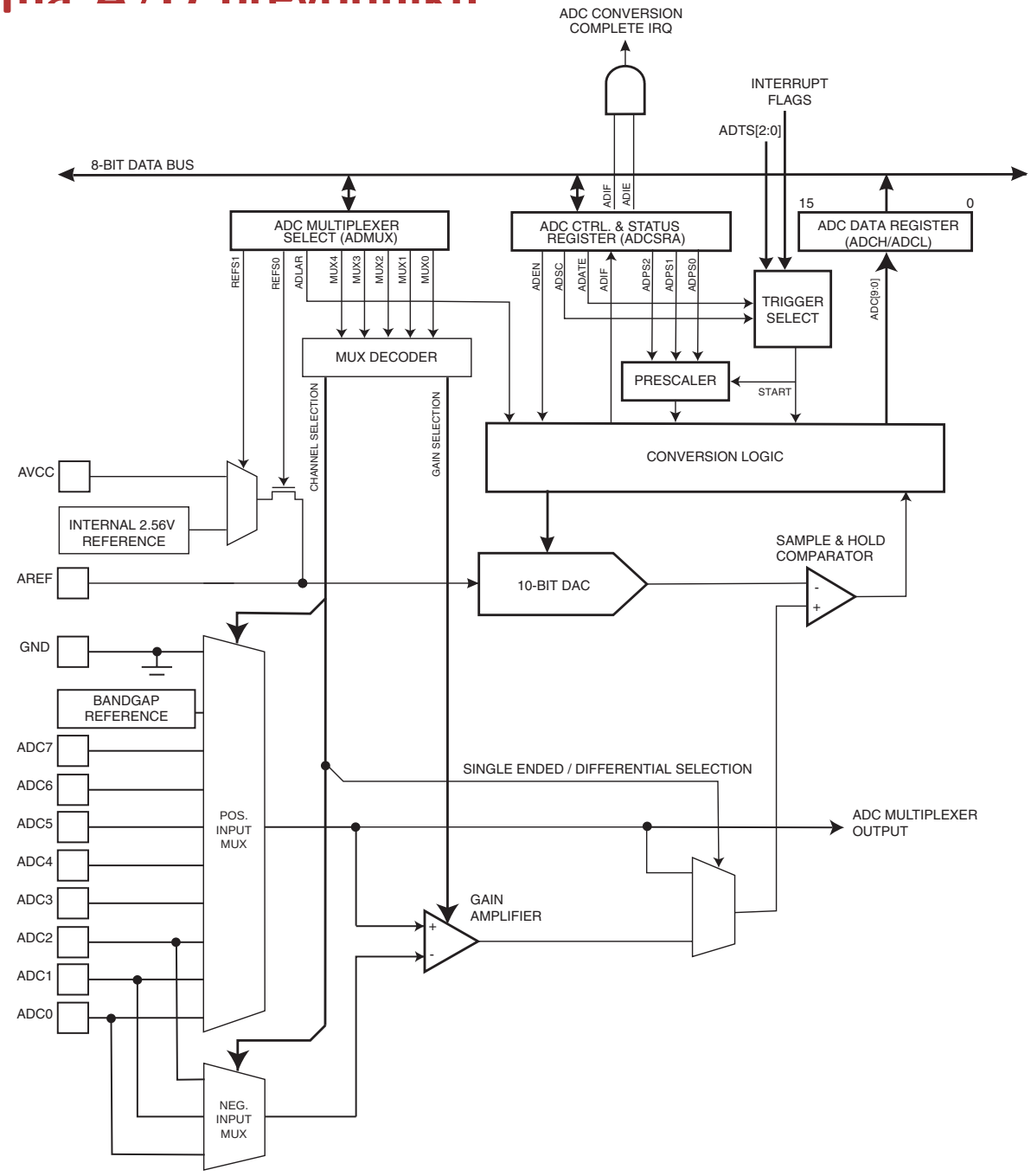
For example purposes, we can assume our DAC or quantiser has 5 bits of resolution and maximum/minimum voltage output swing of +1 and -1 volts. The input/output characteristic is shown below:



A/D převod s pomocí AVR

- MCU umožňuje zpracovávat vstupní analogové signály pomocí A/D převodníku (Analog to Digital Converter) a komparátoru (Analog Comparator).
- ATmega16: 8kanálový, 10bitový převodník – piny PA0 až PA7. Hodnota ve formě dvojkového doplňku (viz přednáška *Číselné reprezentace v MPT*).
- Vstupní úroveň je brána buď vůči *GND*, případně lze převádět diferenční hodnotu dvou vstupních kanálů a to včetně nastavitelného zesílení.
- Minimální hodnota: 0 V. Maximální hodnota: napětí referenčního napětí V_{REF}.
- A/D převodník může pracovat z několika režimech:
  - **jednoduchý převod** (Single Conversion). Převede se hodnota jediného vzorku, poté je převod zastaven,
  - **automatické spouštění** (Auto Triggering). Speciální událost (např. externí přerušení, změna hodnoty komparátoru, přetečení časovače, komparace časovače/čítače, . . .) může zahájit A/D převod. Převod je zahájen náběžnou hranou vybraného signálu; pokud přijde nová náběžná hrana a převod není dokončen, pak je ignorována. (Umožňuje zahájení převodu v konstantních intervalech.),
  - tzv. **volný běh** (Free Running). Převod je prováděn neustále – jeden je ukončen, druhý spuštěn.
- Po dokončení A/D převodu je výsledek zapsán do datových registrů A/D převodníku, tj. do ADCH:L; možnost generovat přerušení – viz vektory přerušení.

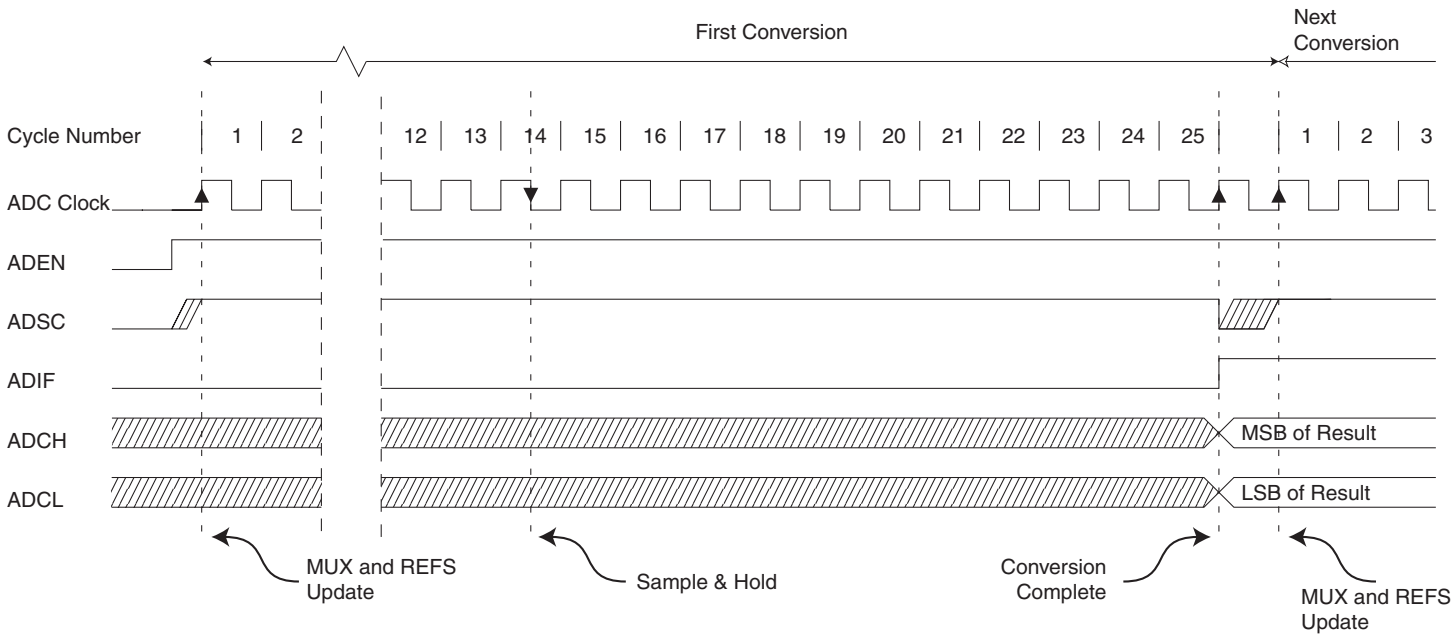
Blokové schéma A/D převodníku



Časování A/D převodníku ATmega16

- interní A/D převodník potřebuje pro 10bitový převod hodinový signál o frekvenci od 50 kHz do 200 kHz:
  - řídicí frekvence může být větší, pokud je postačující nižší rozlišení než 10bitové,
  - A/D převodník obsahuje předdělička (2, 4, 8, 16, 32, 64, 128) ke generování vhodné frekvence, odvozené od frekvence hodinového signálu jádra MCU (f_{CPU}).
- Doba dokončení A/D převodu závisí na zvoleném režimu, přičemž převod první hodnoty trvá déle než všechny následující – inicializace vnitřních obvodů jednotky.
- Celková doba převodu se skládá z doby navzorkování a samotného převodu.

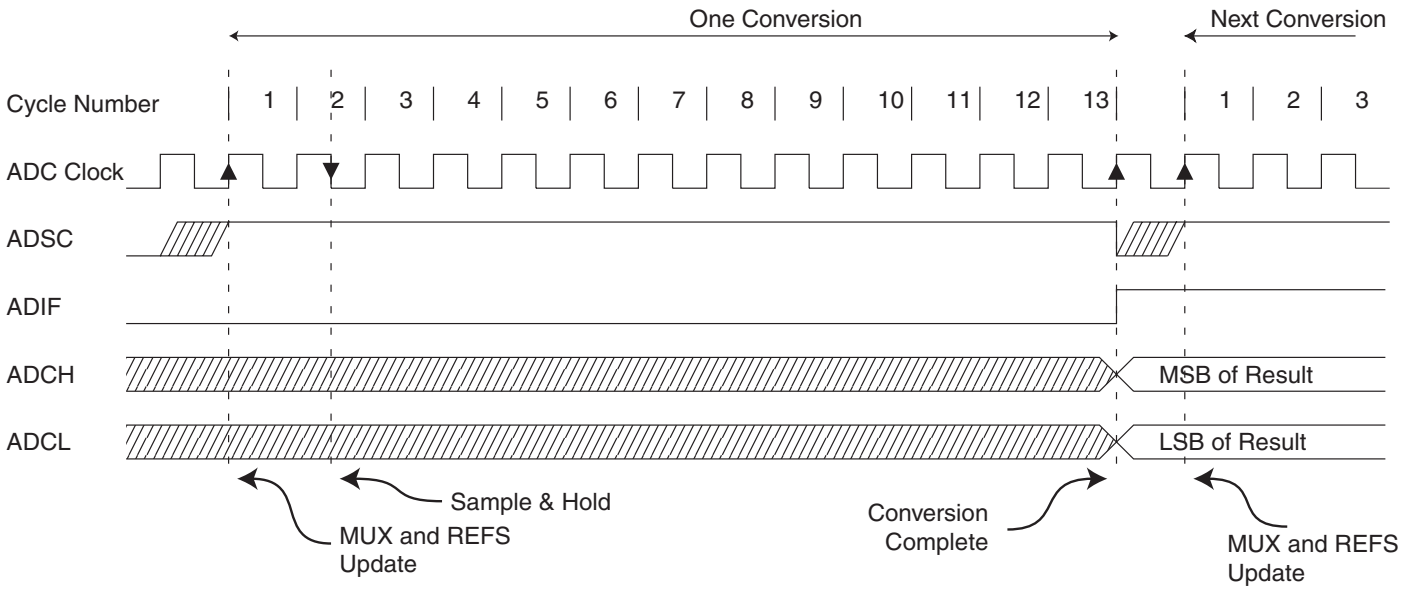
Doba trvání A/D převodu



Obrázek: Doba převodu prvního vzorku.

- Navzorkování prvního vzorku vstupního signálu (sample & hold) trvá 13,5 A/D cyklů.
- Dokončení prvního převodu včetně inicializace analogových obvodů trvá 25 hodinových cyklů A/D převodníku (frekvence 50–200 kHz).

Doba trvání A/D převodu



Obrázek: Doba převodu druhého a následujících vzorků.

- *Navzorkování* ostatních vzorků trvá jen 1,5 cyklů.
- *Dokončení* převodu ostatních vzorků trvá jen 13 cyklů.
- Maximální vzorkovací frekvence jednoho kanálu je tak:
  - 3,8 kSPS při min. frekvenci $f_{ADCmin} = 50\,\text{kHz}$ a
  - 15,3 kSPS při max. $f_{ADCmax} = 200\,\text{kHz}$.

Vyjádření výsledku A/D převodu

- *n*bitový jednoduchý A/D převod konvertuje vstupní napětí lineálně mezi *GND* a V_{REF} v 2^n krocích (tj. hladiny 0 až 2^n-1).
- Tzv. **jednoduchý převod**:

$$\mathrm{ADCH{:}L} = \frac{V_{IN}}{V_{REF}} \cdot (2^n - 1) \tag{1}$$

  - V_{IN}: převáděná hodnota napětí na zvoleném vstupním pinu,
  - V_{REF}: zvolené referenční napětí.

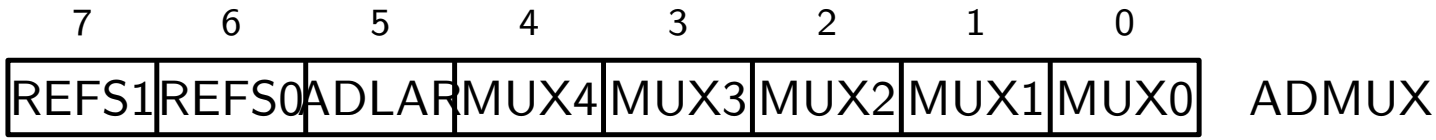
- **Diferenční převod**:

$$\mathrm{ADCH{:}L} = \frac{V_{POS} - V_{NEG}}{V_{REF}} \cdot (2^{n-1} - 1) \cdot GAIN \tag{2}$$

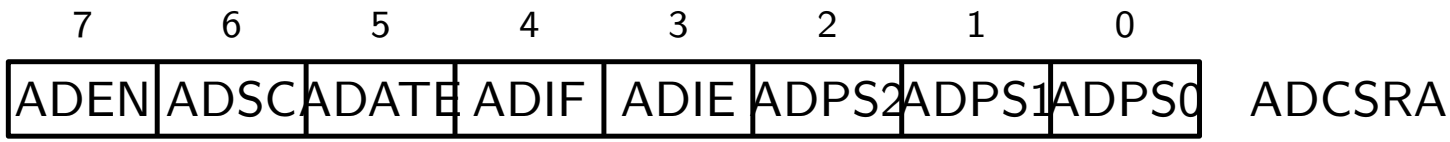
  - V_{POS}: napětí na neinvertujícím vstupním pinu,
  - V_{NEG}: napětí na invertujícím vstupním pinu (viz blokové schéma A/D převodníku),
  - *GAIN*: zvolené zesílení,
  - Výsledek ve dvojkovém doplňku (10 bitů, tj. od -512 do $+511$).

Výběr signálů pro A/D převod – viz blokové schéma ADC

- Lze vybrat různé vstupní a referenční signály pro převod.
- Volby referenčního napětí:
  - externí napětí na pinu AREF,
  - úroveň napájecího napětí,
  - vnitřní zdroj reference 2,56 V.
- Volby vstupních kanálů:
  - individuální vstupy na pinech PA7:0 (ADC7:0),
  - některé předvolené kombinace dvou vstupních kanálů se zesílením 1×, 10×, 100× nebo 200× (viz katalogový list ATmega16).
- Jednotlivé kanály nelze převádět paralelně – vždy je potřeba vybrat jediný vstupní signál (prostřednictvím řídicích registrů). Po dokončení jednoho převodu (využití přerušení): změnit vstupní kanál a zahájit převod na následujícím kanále; viz Ukázka programu pro AVR na konci prezentace.



Obrázek: Struktura kontrolního registru ADMUX (ADC Multiplexer Selection Register).



Obrázek: Struktura kontrolního registru ADCSRA (ADC Control and Status Register A).

Příklad užití A/D převodu

Příklad

Jakou hodnotu obsahuje registrový pár ADCH:L, je-li nastaven diferenční převod vstupů ADC3 = 300 mV a ADC2 = 500 mV, zesílení GAIN = 10× a vnitřní zdroj referenčního napětí V_{REF} = 2,56 V?

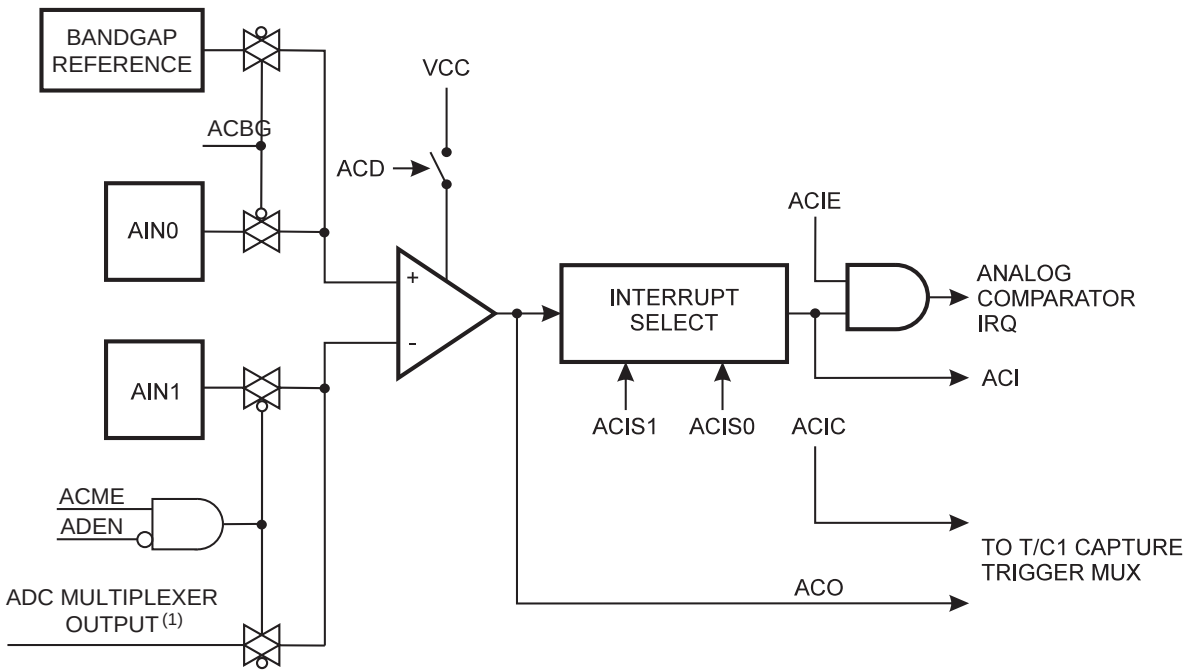
Řešení

$$ADCH{:}L = \frac{V_{POS} - V_{NEG}}{V_{REF}} \cdot (2^{n-1} - 1) \cdot GAIN$$

$$ADCH{:}L = \frac{0{,}3 - 0{,}5}{2{,}56} \cdot (511) \cdot 10 \approx -399 = 0x271$$ *(viz dvojkový doplněk)*

Analogový komparátor u AVR

- Analogový komparátor porovnává neinvertovaný vstup AIN0 s invertovaným AIN1 (u ATmega16 piny PB2, PB3).
- Pokud je úroveň napětí AIN0 $>$ AIN1, výstup komparátoru ACO=1 (Analog Comparator Output). (Synchronizace výstupu komparátoru a ACO bitu trvá 1–2 cykly.)
- Může být generováno přerušení při překlopení komparátoru z 1→0, 0→1, nebo při libovolné změně.



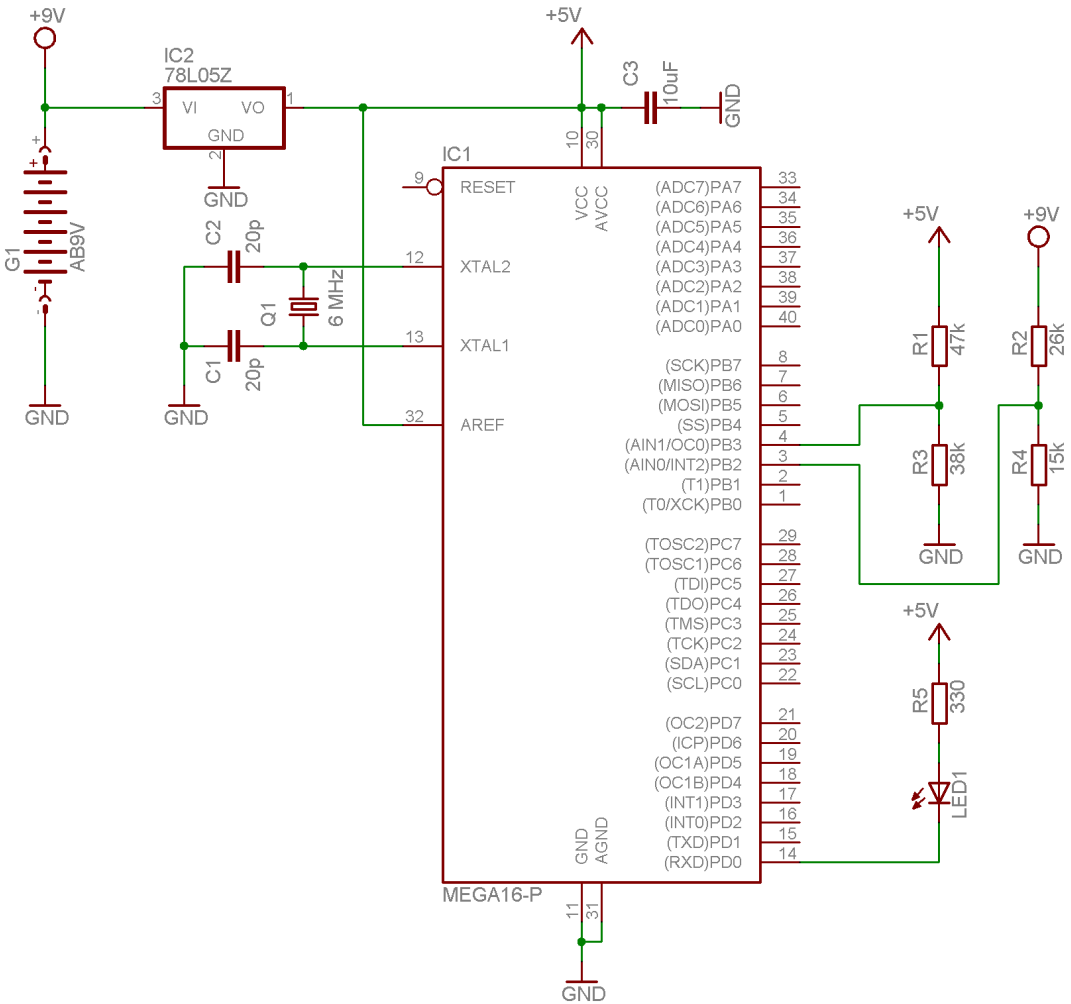
Porovnávané signály

- Analogový komparátor umožňuje kromě porovnávání dvou vstupních signálů z pinů PB2 (AIN0) a PB3 (AIN1), připojení ještě jiných signálů (hodnot).
- Neinvertovaný vstup:
  - externí signál z pinu PB2 nebo,
  - vnitřní zdroj napěťové úrovně o velikosti 1,23 V.
- Invertovaný vstup:
  - externí signál z pinu PB3 nebo,
  - libovolný vstupní kanál A/D převodníku, tj. signály z pinů PA7:0 (ADC7:0). (Konkrétní vstup nastavuje multiplexer A/D převodníku.)

Příklad

Navrhněte obvodové zapojení aplikace, kontrolující napětí na napájecí baterii. Pokles nechť je signalizován blikající LED diodou. Nakreslete vývojový diagram této aplikace.

Příklad užití analogového komparátoru



- Monitorování stavu napájecí baterie pomocí komparátoru.
- Jestliže AIN1>AIN0, výstup komparátoru = 0, LED1 bliká.
- Nastavení odporových děličů:
  - $AIN1 = \frac{R3}{R3+R1} \cdot 5V$,
    $AIN1 = \frac{38}{38+47} \cdot 5V = 2,2V$,
  - $AIN0 = \frac{R4}{R4+R2} \cdot V_{BAT}$,
    $AIN0 = \frac{15}{15+26} \cdot V_{BAT}$,
    $\Rightarrow$ min. napětí baterie $V_{BATmin} = 6V$.
- Obsluha přerušení `ANA_COMP_vect`:
  - vypnutí komparátoru,
  - zahájení blikání LED1 pomocí č/č 0,
  - ...

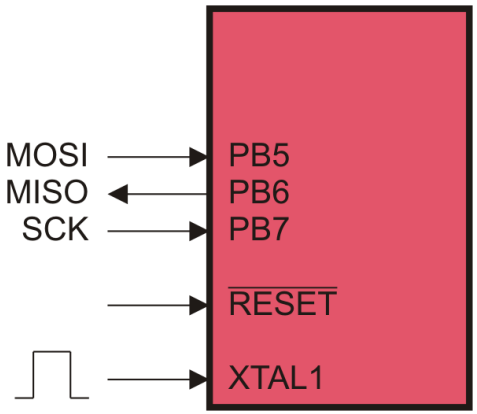
Obrázek: Aplikace, kontrolující napětí na napájecí baterii.

Obsah přednášky

Způsoby programování mikrokontrolérů

- Programování, nebo tzv. download mikrokontrolérů znamená nakopírování vytvořené aplikace (nejčastěji ve formátu Intel HEX) do programové paměti (Flash) mikrokontroléru. (Programátory umožňují také načtení, verifikaci a smazání paměti.)
- Základní způsoby downloadu:
  - paralelní programování,
  - rozhraní JTAG (umožňuje také ladění aplikace v cílovém MCU),
  - sériové programování (ISP In-System Programming) pomocí SPI rozhraní (viz Sériová komunikace).
- Paralelní programování:
  - nutný větší počet vodičů. Zpravidla pomocí externího programátoru – vyjmutí mikrokontroléru ze systému, což je komplikované, krkolomné, někdy nemožné,
  - rychlost downloadu je velká.

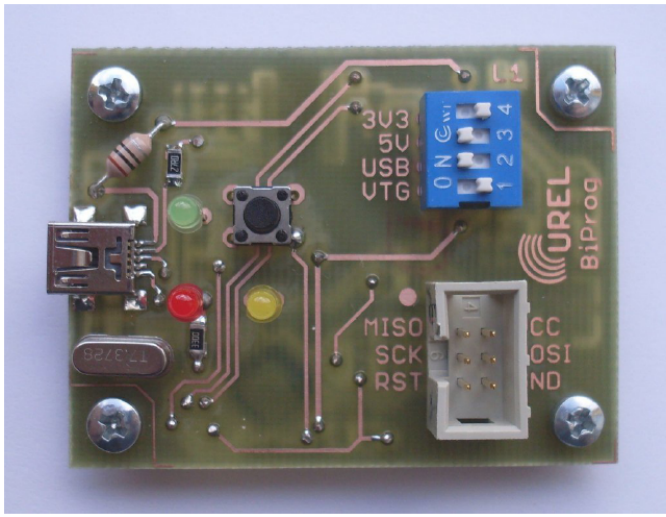
Sériové programování v systému

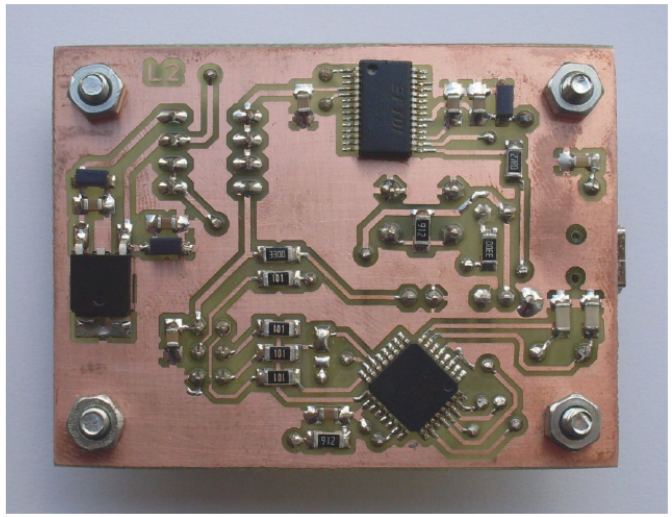


Obrázek: Připojení sériového programování.

- Sériové programování v systému (ISP: In-System Programming):
  - obsahují všechny mikrokontroléry,
  - není nutné vyjímat mikrokontrolér ze systému,
  - neposkytuje možnost ladění aplikace,
  - využívá se komunikace po sériovém rozhraní SPI (Serial Peripheral Interface); viz přednáška Sériové komunikace v MPT,
  - SPI je duplexní, synchronní přenos pomocí čtyř vodičů: *SCK* (hod. signál), *MOSI* (Master Out Slave In), *MISO* (Master In Slave Out), $\overline{SS}$ (Slave Select).
  - Lze programovat programovou a EEPROM paměť. Hodnota na pinu $\overline{RESET}$ je připojena na *GND* (na rozdíl od paralelního programování).

Pozn.: V laboratoři: AVRISP mkII od Atmel.





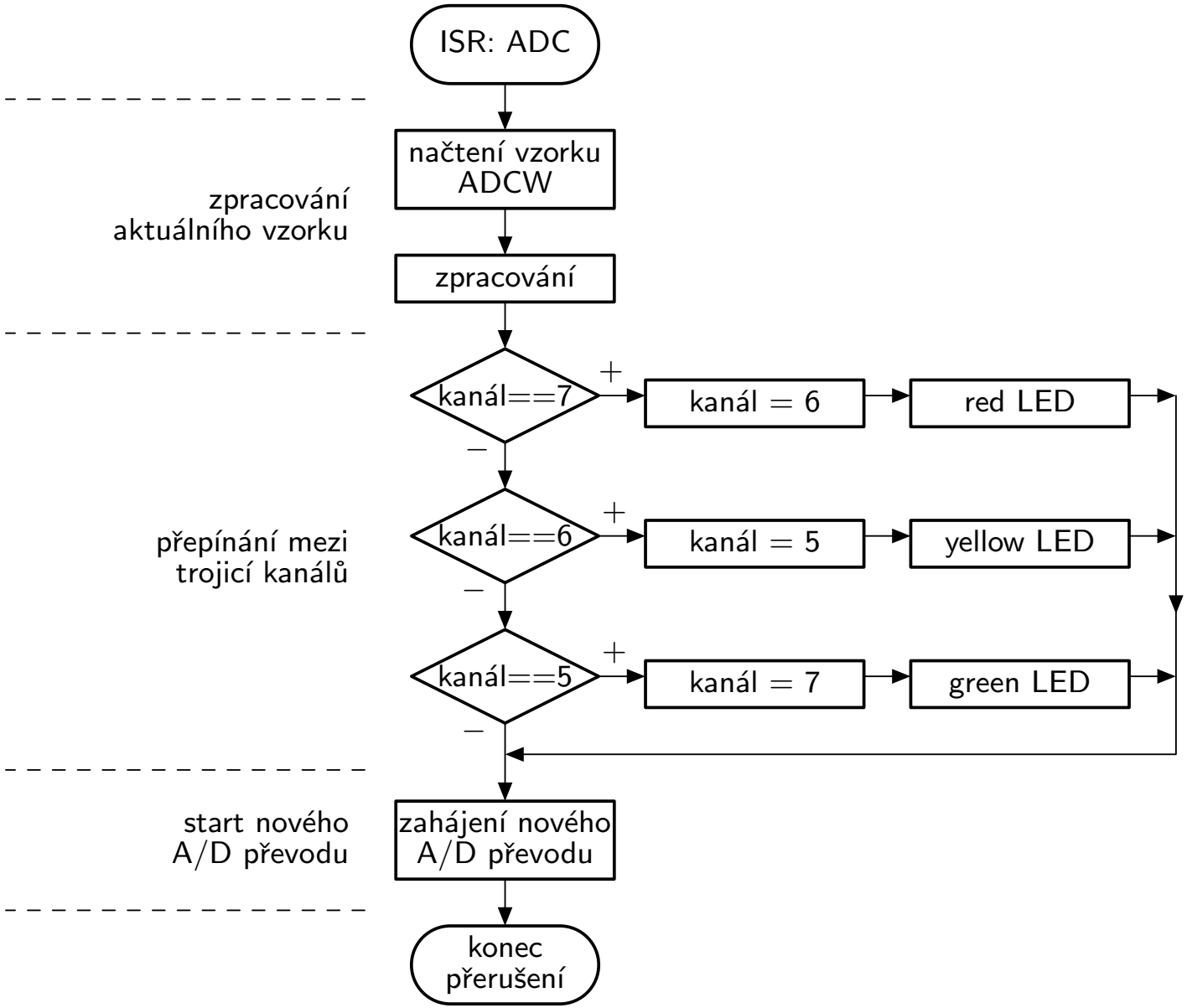
Obrázek: ISP programátor BiProg pro Atmel, Ing. Povalač, `http://www.urel.feec.vutbr.cz/~fryza/?Download`.

Obsah přednášky

Využití A/D převodníku

- Aplikace využívá 3 kanály interního A/D převodníku mikrokontroléru ATmega16. Mezi kanály se přepíná v časovém multiplexu – využívá se globální proměnná *chnl*.
- Řídicí registry:
  - ADMUX (ADC Multiplexer Selection Register) – vybrání jednoho kanálu, příp. referenčního napětí,
  - ADCSRA (ADC Control and Status Register) – zapnutí A/D převodníku, zahájení převodu, povolení přerušení po dokončení převodu, nastavení předděličky hodinového signálu pro A/D převodník.
- Direktivy pre-procesoru `#define` definují řetězce (LEDs, red, ...), které budou před překladem nahrazeny konkrétním příkazem, hodnotou (POTRB, 0b11011111, ...).
- Obsluha přerušení po dokončení převodu `ISR( ADC_vect )`:
  - proměnná *ADCW* obsahuje 10bitovou hodnotu převodu (tj. obsah obou datových registrů ADCH:L); POZOR, proměnnou kterou plníme `ADCW` deklarovat uvnitř obsluhy přerušení, nebo jako `volatile`!!
  - testováním hodnoty proměnné `channel` se provádí podmíněná obsluha tří kanálů; současně se nastavuje následující kanál – registr ADMUX,
  - znovuzahájení převodu – registr ADCSRA, bit ADSC.

Přepínání mezi kanály A/D převodníku



Obrázek: Obsluha přerušení po dokončení A/D převodu. Přepínání mezi 3 kanály interního A/D převodníku s LED signalizací.

A/D převodník v jazyce C

```
1  #include <avr\io.h>          // hlavičkový soubor pro mikrokontrolér
2  #include <avr\interrupt.h>   // hlavičkový soubor pro přerušení
3
4  #define LEDs    PORTB        // direktivy pre-procesoru
5  #define red     0b11011111   // červená LED dioda je připojená na pinu PB5
6  #define yellow  0b10111111   // žlutá LED dioda je připojená na pinu PB6
7  #define green   0b01111111   // zelená LED dioda je připojená na pinu PB7
8
9  char channel = 7 ;           // pomocný identifikátor A/D kanálu
10
11
12 int main( void ){
13
14     DDRB = 0b11100000 ;      // deklarace výstupních pinů pro barevné LED
15
16     // výběr kanálu na pinu PA7
17     ADMUX |= (1<<MUX2) | (1<<MUX1) | (1<<MUX0) ;
18
19     // zapnutí A/D převodníku, zahájení převodu, povolení přerušení
20     // předdělička = 128
21     ADCSRA |= (1<<ADEN)  | (1<<ADSC) | (1<<ADIE) | (1<<ADPS2) | (1<<ADP1) |
22               (1<<ADP0) ;
23
24     sei() ;                  // globální povolení všech přerušení
25     while( 1 ) ;             // nekonečná smyčka
26
27     return( 1 ) ;            // hlavní funkce vrací hodnotu 1
28 }
29 // pokračování na další straně
```

A/D převodník v jazyce C – dokončení

```
30  // pokračování z předešlé strany
31  ISR( ADC_vect ) {           // obsluha přerušení A/D převodníku
32      unsigned int adcData ;
33
34      adcData = ADCW ;        // načtení 10bitové hodnoty A/D převodu
35      ...                     // kód obsluhy přerušení; zpracování vzorku
36
37      // režie přepínání mezi 3 kanály; vizuální signalizace pomocí LED diod
38      if( channel == 7 ){
39          // výběr kanálu na pinu PA6
40          ADMUX |= (1<<MUX2) | (1<<MUX1) | (0<<MUX0) ;
41          LEDs = red ;        // signalizační led
42          channel = 6 ;
43      }
44      else if( channel == 6 ){
45          // výběr kanálu na pinu PA5
46          ADMUX |= (1<<MUX2) | (0<<MUX1) | (1<<MUX0) ;
47          LEDs = yellow ;     // signalizační led
48          channel = 5 ;
49      }
50      else if( channel == 5 ){
51          // výběr kanálu na pinu PA7
52          ADMUX |= (1<<MUX2) | (1<<MUX1) | (1<<MUX0) ;
53          LEDs = green ;      // signalizační led
54          channel = 7 ;
55      }
56
57      // zapnutí A/D převodníku, zahájení nového převodu
58      ADCSRA |= (1<<ADEN) | (1<<ADSC) ;
59  }
```