Úvod do mikroprocesorové techniky

Digitální elektronika 2

doc. Ing. Tomáš Frýza, Ph.D.

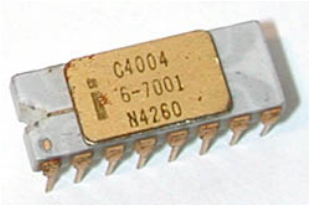
říjen 2020

Obsah přednášky

Obsah přednášky

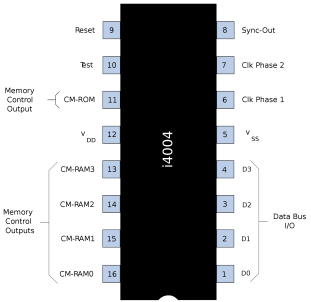
Mikroprocesor

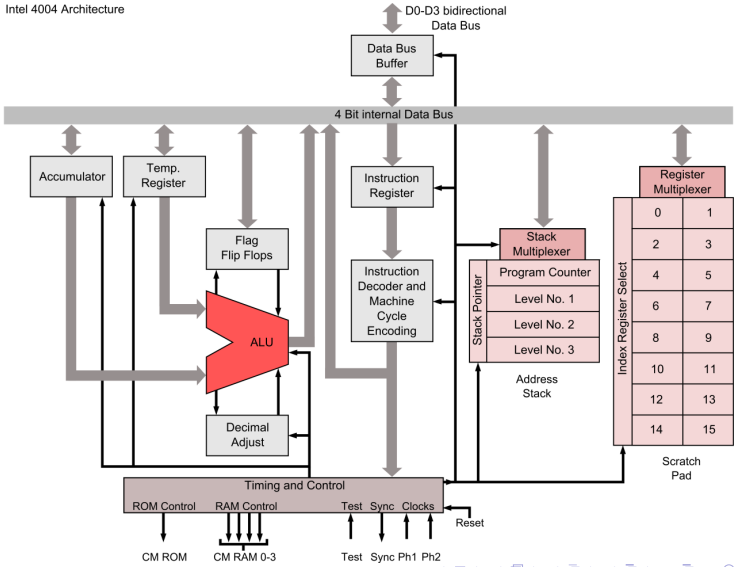
- Mikroprocesor (Microprocessor Unit, MPU) je centrální řídicí jednotka (Central Processing Unit, CPU)
- Jedná se o víceúčelové programovatelné elektronické zařízení řízené hodinovým signálem určené k provádění aritmetických a logických operací pomocí aritmetické logické jednotky neboli ALU.
- Obvykle provádí instruktážní cyklus načítání, dekódování a provádění (fetch, decode, execute)
- Mikroprocesor také komunikuje s dalšími externími součástmi, s paměťovou jednotkou prostřednictvím paměťového rozhraní a může provádět I/O operace na základě konkrétních pokynů.



- První mikroprocesor Intel 4004 z roku 1971
- 4bitová CPU, 16pinové pouzdro, hodinový signál o frekvenci 740 kHz, Harvardská architektura (tj. oddělená paměť pro program a data)
- 16 4bitových registrů
- Instrukční sada pro 46 instrukcí
- cca 2 300 tranzistorů; výrobní technologie: 10 000 nm

Bloková struktura mikroprocesoru Intel 4004





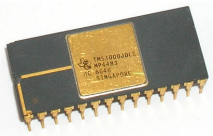
Mikropočítač

- Doplněním mikroprocesoru o podpůrné obvody, tj. vstupně/výstupní periferie a paměť pro program i data vznikne mikropočítač.
- První mikropočítače vznikaly v polovině 70. let, bez klávesnice a displeje. Velikost paměti typicky 4 až 16 kB

- Jedním z prvních mikropočítačů Altair 8800 z roku 1975
- Obsahuje 8bitový mikroprocesor Intel 8080A, hodinový signál 2 MHz
- Velikost paměti RAM 256 B až 64 kB
- Instrukční sada pro 78 instrukcí

Mikrokontrolér

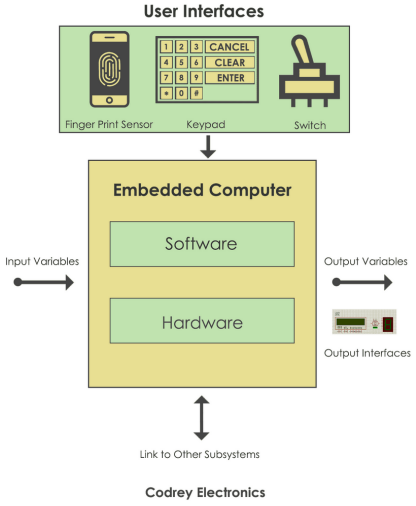
- Mikrokontrolér (Microcomputer Unit, MCU) se skládá z mikroprocesoru a dalších komponent, jako jsou paměťové bloky, digitální I/O, analogové I/O, časovače a další základní periferie.
- Mikroprocesor uvnitř MCU je obvykle jednojádrový procesor a optimalizuje cena/výkon. On-chip paměti jako RAM (Random-access memory) a ROM (Read-only memory) jsou také optimalizovány z hlediska rychlosti, nákladů, trvanlivosti a energetické náročnosti.
- Velké množství výrobců i dodávaných řad mikrokontrolérů: Intel, NXP (dříve Freescale), Microchip Atmel, STM, ...
- Mezi další hardwarové komponenty mohou patřit různé vstupní a výstupní komponenty, časovače, struktury přerušení, externí (sériové) sběrnice, ADC, atd. Ty jsou vybírány na základě konečného účelu. Všechny komponenty v mikrokontroléru komunikují prostřednictvím interní systémové sběrnice.



Obrázek: Mikrokontrolér TMS1000 firmy Texas Instruments

- První mikrokontroléry vytvořila firma Texas Instruments pod označením TMS1000 v roce 1974
- 28pinové pouzdro, hodinový signál o frekvenci 400 kHz, 4bitová sběrnice
- velikost paměti RAM 32 B, ROM 1 kB
- 4 vstupní piny, 11 výstupních, 8bitový výstupní paralelní port

Embedded systémy



- Embedded systém (vestavěný systém) je specializovaný výpočetní systém s hardwarem, který je založen na čipu (může to být mikrokontrolér nebo mikroprocesor). Jeho integrovaný hardware se bude lišit v závislosti na tom, k čemu je systém navržen.
- Obvykle jsou navrženy k provádění přesně definovaného úkolu, který se opakuje buď periodicky, nebo po aktivaci. Tyto systémy mohou interagovat s většími systémy a okolním prostředím.
- Vestavěné systémy mají nízkou spotřebu energie, relativně vysoký výkon a jsou levné.
- Vestavěné systémy musí často pracovat v reálném čase. Takový systém zaručuje smysluplný výstup vytvořený během definovaně krátké doby nebo intervalu.
- Obvykle obsahuje jeden nebo více vzájemně propojených MCU.
- Většina součástí embedded systému jsou elektrické součásti. Důležitou funkčnost však může poskytnout také mechanická konstrukce nebo umístění snímačů.

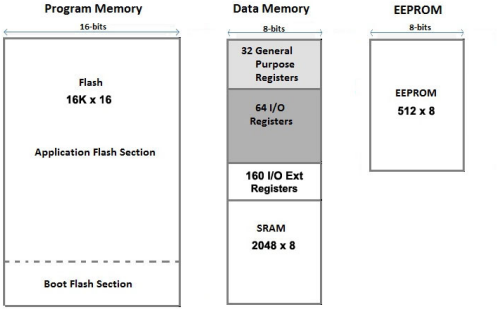
Obsah přednášky

Základní dělení mikropočítačů podle architektury

- První dělení mikropočítačů iniciovala americká vláda v 70. letech, když požádala Princetonskou a Harvardskou univerzitu, aby navrhly architekturu vhodnou pro potřeby dělostřelectva
- Vznikly dvě základní koncepce:
  - Von Neumannova
  - Harvardská
- Von Neumannova architektura:
  - popisuje jak má číslicový systém pracovat a z jakých hlavních částí by se měl skládat: řídicí jednotka, paměti, I/O obvody
  - zásadní myšlenka von Neumannovy architektury je použití pouze jedné paměti a to pro kontrolní program (instrukce) i pro data (proměnné, . . .) – obojí je "jedno a totéž"!
  - nekoresponduje s vyššími programovacími jazyky; např. neumožňuje pracovat s vícerozměrnými poli. Von Neumannova architektura např. v počítačích PC
  - program je vykonáván sekvenčně, tj. instrukce se provádějí tak jak jdou za sebou – "až na ně dojde řada"
  - vnitřní architektura je nezávislá na řešené úloze. Veškeré změny mají být řešeny softwarově, tzn. počítač je řízen obsahem paměti
  - původní přednost v univerzálnosti architektury je ve svém důsledku nevýhodná – systém dokáže zpracovat libovolný problém, ale neefektivně
  - paměť je rozdělena na stejně velké buňky, jejichž pořadové čísla se využívají jako identifikační adresy

Harvardská architektura

- Harvardská architektura chronologicky navazuje na architekturu von Neumannovu a mění některé její vlastnosti
- Zásadní rozdíl je oddělená část paměti pro program a data
- možnost použití pamětí odlišných technologií (EEPROM, Flash, ...)
- dvě sběrnice (pro instrukce, pro data) umožňují současný přístup k instrukcím i k datům
- nevyužitou část paměti pro data ovšem nelze využít pro uložení programu a naopak
- Sekvenční vykonávání instrukcí zachováno



Obrázek: Koncepce oddělené paměti Harvardské architektury u 8bitového mikrokontroléru ATmega328

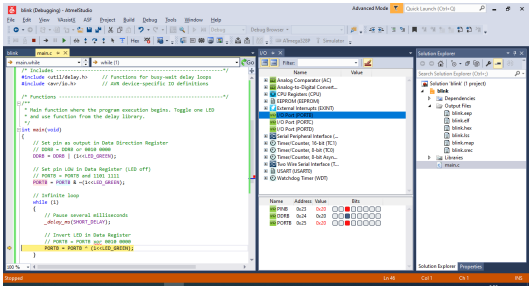
Procesory CISC/RISC

- Dosavadní dělení procesorů výlučně podle hardwaru. Dále základní dělení procesorů z pohledu instrukční sady:
  - CISC (Complex Instruction Set Computer – Počítač s komplexním souborem instrukcí)
  - RISC (Reduced Instruction Set Computer – Počítač s redukovaným souborem instrukcí)
- CICS procesory obsahují velké množství instrukcí, které s malými obměnami vykonávají ty samé operace (např. pomocí přímého adresování, indexového adresování, apod.) – lze je snadno nahradit posloupnosti jiných instrukcí
- Výskyt některých instrukcí je velmi nízký $\Rightarrow$ proč mít tyto instrukce v instrukčním souboru? Každá instrukce rozšiřuje složitost části procesoru určené pro dekódování (rozpoznání) instrukce.
- Procesory RISC se kromě malého počtu instrukcí vyznačují také:
  - malým počtem způsobů adresování
  - používá zřetězené zpracování instrukcí
  - instrukce mají pevnou délku (u AVR 16 bitů) a jednotný formát, což urychluje jejich dekódování
  - používají větší počet rovnocenných registrů (u AVR 32 reg. R0, R1, . . ., R31)
- Výsledný program pro procesory RISC:
  - je zpravidla delší z důvodu většího počtu instrukcí s konstantním počtem bitů
  - doba vykonání programu může být kratší, protože většina instrukcí se vykoná v jednom hodinové cyklu

Obsah přednášky

Vývoj aplikací, simulátory

- Postup při vývoji aplikací: vytvoření zdrojového kódu aplikace a její odladění v simulátoru, příp. emulátoru; pokud jsou v cílové aplikaci i jiné hardwarové periférie, je vhodné vyzkoušet též na vývojové desce



Obrázek: AtmelStudio 7

Simulátor

- Simulace chování programu na jiném než cílovém procesoru
- Simuluje se přeložený kód, tj. musí být k dispozici převod zdrojového kódu do strojového jazyka požadovaného cílového obvodu
- Obsahuje spouštěcí a ladící programy: krokování programu, breakpointy, ...
- Zpravidla omezené možnosti simulace okolního (hardwarového) prostředí
- Nepracuje v reálném čase; zpravidla je možné získat informaci o době výkonu programu v počtech taktů hodinového signálu

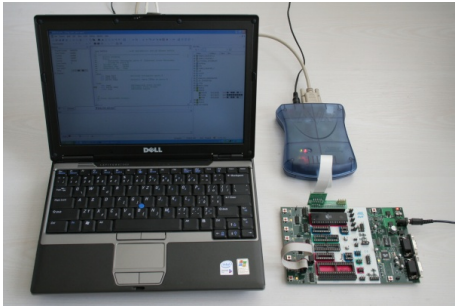
Vývoj aplikací, emulátory, vývojové desky

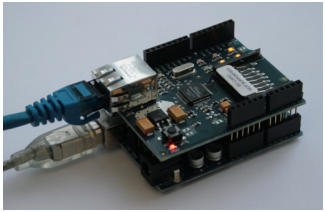
Emulátor

- Propojení počítače s cílovým procesorem
- Obslužný software (na PC) obsahuje stejné náležitosti jako u simulátoru, ale odlišné spouštěcí a ladící nástroje
- Umožňuje monitorování cílového procesoru během ladění (obsah proměnných, pozice v paměťovém prostoru, ...)

Vývojová deska

- Hardwarové zařízení (mimo počítač) umožňující odladění aplikace včetně připojení základních periferií (displej, USB, relé, ...)
- V závislosti na aplikaci, není potřeba vytvářet finální zapojení před odladěním; univerzální deska usnadňuje vývoj nové aplikace

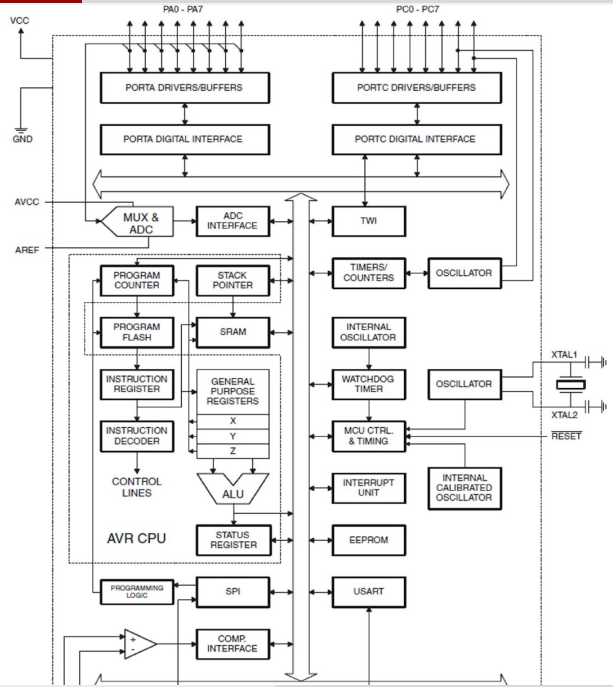




Obrázek: Vývojová deska STK500 a Arduino s Ethernetovým modulem

Microchip Atmel AVR

AVR XMEGA MCU	Xtreme performance 8-bit	General Purpose Lighting LCD	picoPower SleepWalking DMA Event System EEPROM Self-programming	8-384KB Flash 32-100 pins Up to 32 MHz 1.0 MIPS/MHz
Details				
megaAVR MCU	More peripherals and options	General Purpose Lighting LCD	QTouch PTC (Peripheral Touch Controller) picoPower SleepWalking EEPROM Self-programming	4-256KB Flash 28-100-pins Up to 20 MHz 1.0 MIPS/MHz
Details				
8-bit tinyAVR MCU	Small and powerful	General Purpose Lighting Basic Motor Control	PTC (Peripheral Touch Controller) QTouch EEPROM 0.7V operation	0.5 – 16KB Flash 6-32 pins Up to 20MHz 1.0 MIPS/MHz



Programátorský model, mikrokontrolér AVR

Definice (Programátorský model)

Programátorský model mikrokontroléru (nebo také softwarový pohled na MCU) obsahuje popis paměťového prostoru MCU, soubor registrů, jejich názvy, adresy a funkce, které může programátor využívat.

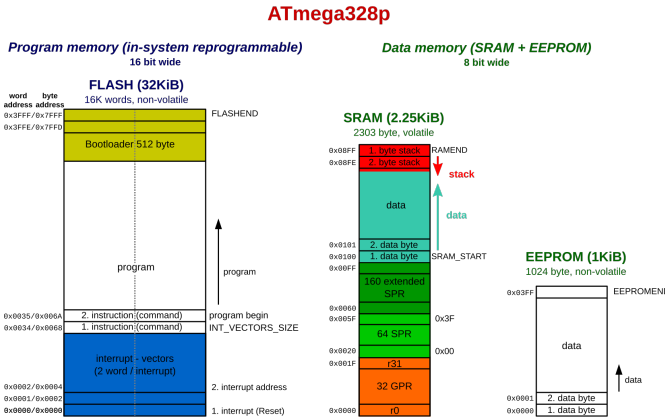
7		0
$00	R0	
$01	R1	
$02	R2	
$03	R3	
$04	R4	
$05	R5	
$06	R6	
$07	R7	
$08	R8	
$09	R9	
$0A	R10	
$0B	R11	
$0C	R12	
$0D	R13	
$0E	R14	
$0F	R15	

7		0
$10	R16	
$11	R17	
$12	R18	
$13	R19	
$14	R20	
$15	R21	
$16	R22	
$17	R23	
$18	R24	
$19	R25	
$1A	R26	X Low
$1B	R27	X High
$1C	R28	Y Low
$1D	R29	Y High
$1E	R30	Z Low
$1F	R31	Z High

- Obecné pracovní registry (anglicky: General Purpose Registers, GPRs). Konečný počet paměťových míst v jádře procesoru s krátkou dobou přístupu
- Pro mikrokon. AVR: 32 8bitových registrů; typická ALU operace vyžaduje dva operandy z GPR, operace se vykoná, výsledek se uloží opět do registru, vše v jednom CPU cyklu
- Názvy registrů: R0, R1, ..., R31
- Adresně namapovány na začátek datové paměti SRAM
- Šest posledních registrů lze využít jako tři 16bitové ukazatele X, Y a Z

Kontrolní registry AVR, Special-Purpose Registers

- Kontrolní registry (Special-Purpose Registers, I/O registers) slouží k řízení jednotlivých periférií i samotného jádra mikrokontroléru: řízení I/O portu, obsah časovače, řízení přerušení, ukazatel na zásobník, stavový registr, ...
- POZOR: Na rozdíl od GPRs neslouží k ukládání "dat", ale každý bit má specifický význam, viz katalogový list/manuál od výrobce
- SPR jsou umístěny v SRAM hned za GPR (pracovními registry)

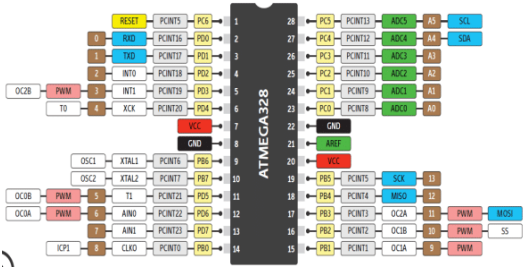


Obsah přednášky

Vstupně/výstupní port

- Nejvšestrannější input/output zařízení mikrokontroléru. GPIO (General Purpose Input/Output)
- Počty pinů korespondují s použitím mikrokontroléru: 4 piny (RS08KA, Freescale), 6 pinů (ATtiny12, Atmel), 32 pinů (ATmega16, Atmel), 48 pinů (ATmega103, Atmel), ...

- ATmega328 obsahuje 3 I/O porty (tj. osmici pinů) s označením B, C a D
- U mikrokontrolérů AVR jsou ke každému obousměrnému portu asociovány 3 kontrolní registry:
  - směrový registr DDRx (Data Direction Register)
  - datový (výstupní) registr PORTx (Data Register)
  - vstupní piny PINx (Port Input Pins)



Obrázek: Pinout mikrokontroléru ATmega328

Ovládání obousměrného portu

- Směrový registr DDRxn (x=A,B,C,D; n=0,1,. . .,7):
  - hodnota 0 ↔ pin je definován jako vstupní
  - hodnota 1 ↔ pin je definován jako výstupní
  - v každém portu je možné libovolně kombinovat vstupní/výstupní piny

7	6	5	4	3	2	1	0	
DDC7	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0	DDRC

Obrázek: Struktura směrového registru DDRC (Port C Data Direction Register)

- Výstupní registr PORTxn :
  - v případě definovaného výstupního pinu, udává hodnota bitu v registru logickou úroveň na pinu. Hodnota 1 (DDRxn=1) ↔ pin je na vysoké úrovni, apod.
  - v případě vstupního pinu, udává hodnota bitu v registru aktivaci/deaktivaci pull-up rezistoru, viz Tabulka

Tabulka: Definice vstupně/výstupního portu

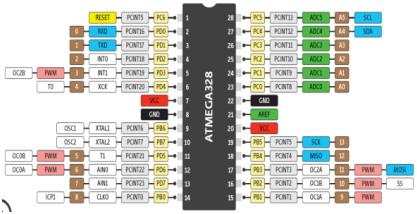
DDRxn	**PORTxn**	**I/O**	**Popis funkce**
0	0	Vstupní	Vysoká impedance
0	1	Vstupní	Aktivace pull-up rezistoru
1	0	Výstupní	Nízká úroveň
1	1	Výstupní	Vysoká úroveň

Ovládání obousměrného portu, pokračování

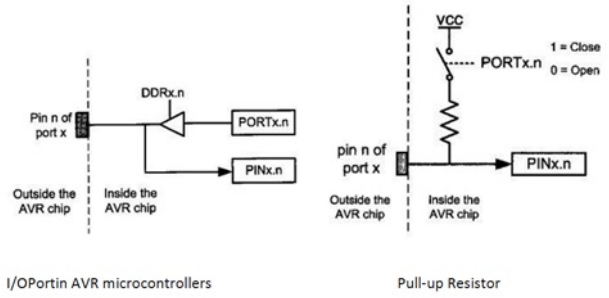
- Vstupní registr PINxn:
  - hodnota 0 nebo 1 korespondující s úrovní signálu na vstupu
  - kopírují se zde také data zapsaná do výstupního registru PORTxn
- Každý pin obsahuje přepěťovou ochranu (omezující diody) a možnost připojení pull-up rezistoru
- Piny umožňují také alternativní funkci, kdy jsou využívány interními periferiemi MCU (PWM od časovače, analogový signál pro ADC, ...)

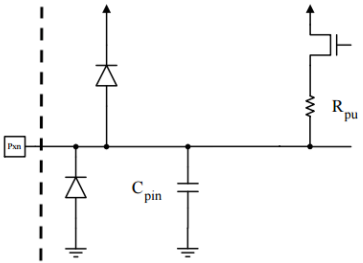
Tabulka: Některé alternativní funkce pinů ATmega328

Pin	Alternativa	Popis funkce
PC5:0	ADC	Vstup 10bitového A/D převodníku
PB5:2	SPI	Sériová komunikace pomocí SPI
PC5:4	TWI	Dvouvodičová sériová sběrnice I2C
PD2	INT0	Zdroj externího přerušení 0
PD1:0	USART	Vstup/výstup sériové komunikace USART



Vnitřní zapojení jednoho I/O pinu





Obrázek: Význam řídicích registrů DDR, PORT, PIN pro jeden I/O pin a přepěťová ochrana pinu

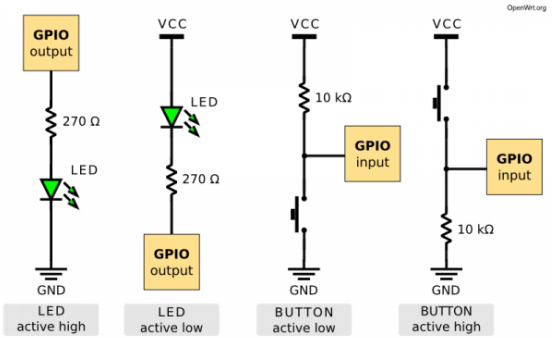
Připojení externích periférií

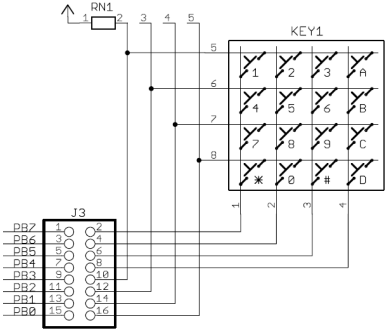
- Připojení externích periférií (tlačítko, LED, klávesnice, ...) je obecně možné realizovat třemi způsoby:

  (1) přímé připojení – jeden pin je využit k připojení jednoho jednoduchého zařízení (tlačítko, LED, ...)

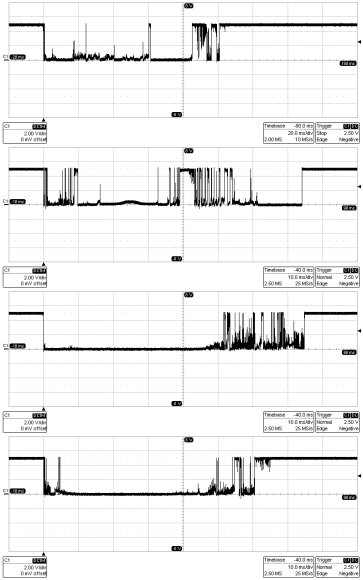
  (2) maticové uspořádání – několik jednoduchých zařízení je uspořádáno do matice, což snižuje počet potřebných I/O pinů (maticová klávesnice)

  (3) pomocí pomocných obvodů/řadičů: externí zařízení pracuje nezávisle na řídicím mikrokontroléru. V případě události generuje požadavek na přerušení, které zajistí načtení/vyslání potřebných dat





Zákmity mechanického tlačítka



Obrázek: První průběh: 20 ms/div, ostatní průběhy: 10 ms/div

- Při sepnutí mechanického tlačítka dochází k zákmitům (nechtěné impulsy způsobující nekorektní výkon programu), které je nutné ošetřit

Pomocí zpožďovací smyčky; blokující

- Při stisku i uvolnění tlačítka volat funkci zpoždění a následně testovat logickou úroveň vstupního signálu

Periodické čtení; jednoduchá implementace

- V periodických intervalech (např. 4 ms) číst stav tlačítka. Pokud posloupnost alespoň tří nul, tlačítko považovat za stisknuté

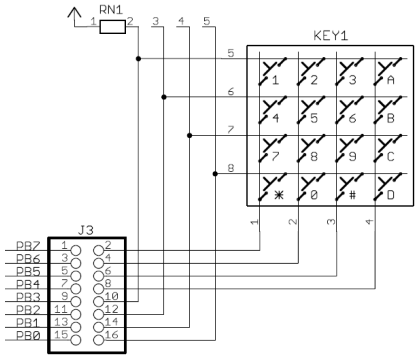
Externí přerušení a časovač

- Při externím přerušení nulovat časovač a s následným přetečením testovat logickou úroveň vstupního tlačítka

- Testovat vždy oba stavy, tj. stisk ale i uvolnění tlačítka

Použití maticové klávesnice

- Nechť maticová klávesnice 4×4 využívá jeden port:
  - 4 piny jsou definovány jako vstupní (např. PB3–PB0)
  - 4 piny jsou výstupní (např. PB7–PB4)
- Zjištění stisknuté klávesy se provádí tzv. skenováním klávesnice, kdy jsou ve čtyřech krocích vyslány hodnoty na výstupní piny:
  (1) PB7:4 = 0b1110
  (2) PB7:4 = 0b1101
  (3) PB7:4 = 0b1011
  (4) PB7:4 = 0b0111
- Po každém kroku se skenují vstupní piny. Pokud je některý roven 0, bylo stisknuto příslušné tlačítko
- Kód tlačítka se vypočte ze "souřadnic", tj. ze sloupce a řádky, které obsahují hodnotu nula
- Proces skenování může probíhat opakovaně, např. v nekonečné smyčce



Obrázek: Připojení maticové klávesnice k portu mikrokontroléru

Obsah přednášky

Vybrané funkce AVR-GCC, knihovny avr-libc a jazyka C

- Testování jednoho bitu v kontrolním registru:

```
if (bit_is_set(PINA, 0)){
    // only if PINA0 = 1
}

if (bit_is_clear(DDRB, 5)){
    // only if DDRB5 = 0
}
```

- Cyklus s testováním bitu v kontrolním registru:

```
loop_until_bit_is_set(PORTC, 2);
// stay here until PORTC2 = 1

loop_until_bit_is_clear(DDRA, 7);
// stay here until DDRA7 = 0
```

- Použití knihovny pro delay:

```
#include <avr/io.h>
#define F_CPU 16000000UL    // Clock frequency
#include <util/delay.h>

...
_delay_ms(100);             // Wait for 100ms
_delay_us(25);              // Wait for 25us
```

- Nastavení/nulování jednotlivých bitů v registru:

```
// set bits to high
PORTB |= (1<<PB7) | (1<<PB0);
PORTB |= _BV(PB7) | _BV(PB0);

// set bits to low
PORTB &= ~((1<<PB7) | (1<<PB0));
PORTB &= ~(_BV(PB7) | _BV(PB0));

// read control register
temp = PINB;
```

- Opakování bitových/logických operací v jazyce C:

  << ... bitový posun doleva

  & ... logický součin AND

  | ... logický součet OR

  ^ ... exkluzivní součet XOR

  ~ ... negace

  _BV() ... makro pro nastavení 1 bitu v bytu