PRAXE

4	**SPŠ Chomutov**	**Kapusňák Jiří**
22.3.2022	**Generátor**	**V4**

Zadání

Vytvořte program, který bude generovat signály na osciloskopu. Program musí obsahovat:

- Zvolení amplitudy
- Zvolení frekvence
- Zvolení posunu
- Zvolení kanálu
- Zvolení jednoho ze tří signálu (sin, tri a squ)

Teorie

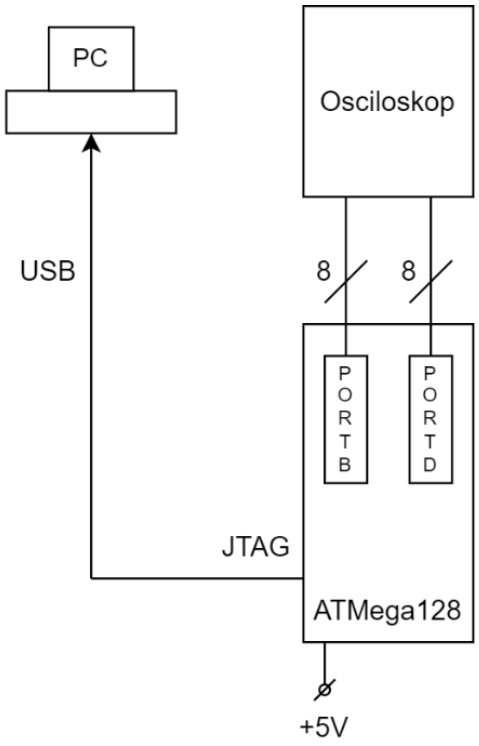
Mikrokontrolér propojíme se osciloskopem pomocí 2 8bitových kabelů pro každý kanál zvlášť, kde PORTB propojíme s kanálem 1 a PORD propojíme s kanálem 2, počítač propojíme s mikrokontrolerem pomocí programátoru JTAG. Dále budeme potřebovat software, který nám překompiluje náš kód do kódu zdrojového, v našem případě Programmers Notepad.

V našem projektu musíme vytvořit MakeFile, kde uchováváme veškeré nastavení našeho mikrokontroléru, ten vytvoříme pomocí programu mfile. Do MakeFilu musíme nastavit náš správny procesor(atmega128), námi zvolený programátor (jtagmk1), frekvenci mikrokontroleru(14745600Hz) a jako poslední, port v počítači ve kterém máme zapojený programátor.

Osciloskop funguje na bázi DAC převodníku. Kde vezmeme bitové slovo a převedeme ho na analogový signál. Jako příklad 0 bitů na log. 1 by nám zobrazilo 0V zatímco všech 8 bitů na log. 1 by nám zobrazilo průběh v 5V (Jelikož napájíme na 5V, což je taky maximum viz. Obrázek č.1)

Pomocí příkazu DDR (Data Direction Register) nastavuje pro námi zvolených 8 bitů, zda bity jsou vstup či výstup. V našem případě nastavujeme oba porty na výstup aby jsme mohli do nich zapisovat.

Schéma zapojení



(obrázek č. 1)

Popis programu:

Po zadání hodnot (frekvence, amplituda a posun) musíme hodnoty převést z reálných hodnot na bity, tím docílíme tak, že si vypočítáme kolik voltů obsahuje pouhá 1. Musíme si zvolit maximální číslo, které budeme do generátoru posílat, tak aby nám signál nešel do saturace, v našem programu jsme si zvolili 210 což jsou přesně 4V.

Převodník tedy funguje na bázi, uživatel zadá, že jeho amplituda jsou 4V což je tedy maximum, máme vypočítáne, že 1 je 0,019V. Vynásobíme tedy 4V číslem 0,019 a dostaneme bitové číslo, což je 210.

Obdelníkový signál funguje jednoduše, podle převedené frekvence v ms, obdelník čeká, poté se přepné do maximalního stavu (nastavená amplituda), opět počká podle převedené frekvence v ms a nastaví se opět na nulu.

U trojúhelníkového signálu si vypočítáváme krok, tak aby nám trojúhelník nezměnil amplitudu při změně frekvence, jeden krok je tedy Amplituda / frekvence, což znamená že frekvence je i počet kroků co musí udělat než dojde do maximální vychýlky a opět počet kroků než dojde zpět na 0.

Pila funguje téměř stejně jako trojúhelníkový signál, akorát její krok musí být 2x menší než trojúhelníkový a při dosažení maximální výchylky spadne hned na nulu a signál se opakuje.

Pro vytvoření sinusového signálu musíme vyjít ze vztahu y = Amplituda/2 * (2*π*f*t + posun)*Amplituda/2.První amplitudou zvolíme maximum, ale nemůžeme zapomenout, že sinus by jel i do mínusu, takže kdyby jsme zvolili amplitudu 210, tak by sinus jel od 0 do 105 do -105 do 0, proto na konci přičítáme tu samou amplitudu znova aby jsme sinus posunuli nahoru a dostali jsme sinus který začíná v 105 jede do 210, pokračuje do 0 a skončí v 105 opět.

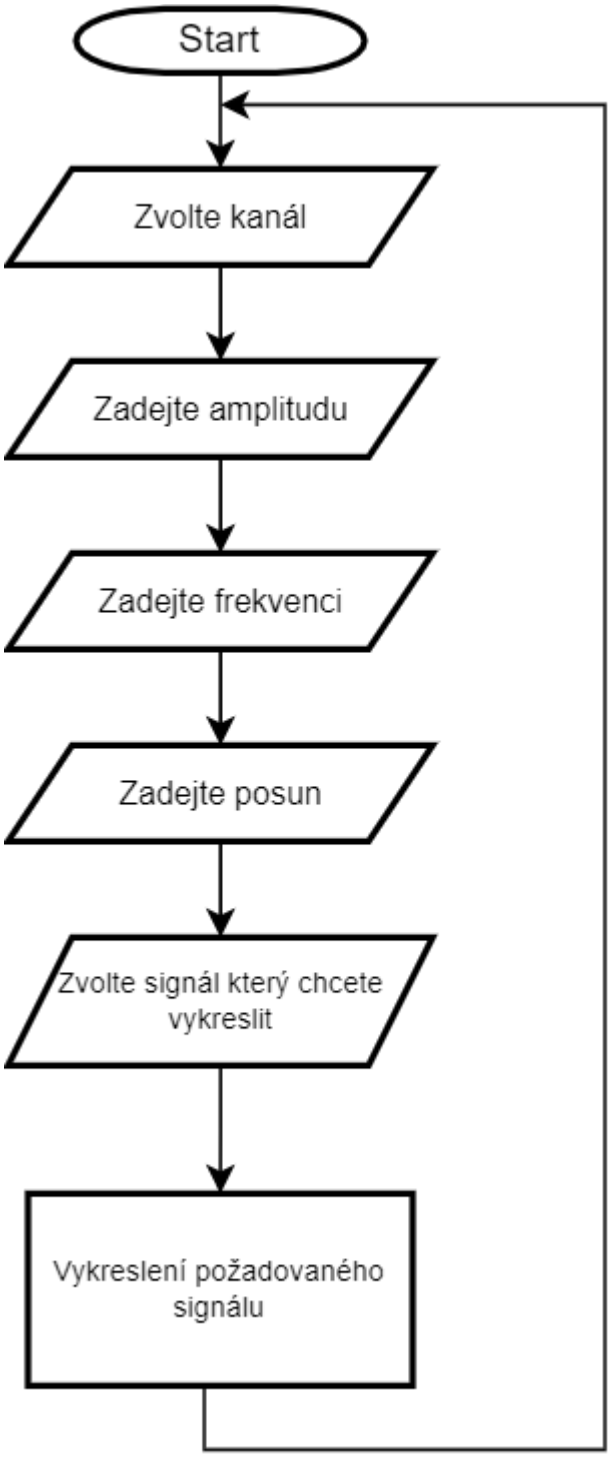
Každá metoda vytvoří vždy posun jen o jeden krok, v našem mainu tyto kroky furt opakujeme s delayem 1ms.

Popis proměnných:

Typ	Argumenty	Metoda	Účel
void		nastaveniportu	Nastaveni portu na výstup
static void		nastavenifrekvence	Převedeni frekvence z Hz na ms
static void		nastaveniamplitudy	Převedeni amplitudy z V na bit
static void		nastaveniposunu	Převedeni posunu z V na bit
static void		nastavenikroku	Nastaveni kroku
static void	int kanal	zapsaniobdel	Nastaveni obdel. Signalu
static void	int kanal	zapsanitroj	Nastaveni troj. Signalu
static void	int kanal	zapsanipila	Nastaveni pily
static void	int kanal	zapsanisin	Nastaveni sin. Signalu

Typ	Proměnná	Účel
int	kroksinch1	Krok sinusu na kanale 1
int	kroksinch2	Krok sinusu na kanale 2
int	sinusch1	Jeden bod sinusoveho signalu na kanale 1
int	sinusch2	Jeden bod sinusoveho signalu na kanale 2
double	cisloch1	Cislo ktere zapisujeme na PORTB (kanal 1)
double	cisloch2	Cislo ktere zapisujeme na PORTD (kanal 2)
double	maxch1	max vychylka kde muze PORTB jit tak aby nesel do saturace(4V)
double	maxch2	max vychylka kde muze PORTD jit tak aby nesel do saturace(4V)
double	cekanich1	hodnota pro cekani mezi 0 a amplitudou pri obdelnikovym signalu (kanal 1)
double	cekanich2	hodnota pro cekani mezi 0 a amplitudou pri obdelnikovym signalu (kanal 2)
int	bitposunch1	bitovy posun kanalu 1
int	bitposunch2	bitovy posun kanalu 2
int	amplitudabitch1	amplituda ch1 v bitovym cisle
int	amplitudabitch2	amplituda ch2 v bitovym cisle
double	frekvencemsch1	Frekvence ch1 v ms
double	frekvencemsch2	Frekvence ch2 v ms
double	p1	Promenna pro krok trojuhelniku a pily na kanalu 1
double	p2	Promenna pro krok trojuhelniku a pily na kanalu 2
double	posunch1	(NAPETI!, MAX 4V) hodnota díky které můžeme posunout začátek signálu kanálu 1
double	amplitudach1	(NAPETI!, MAX 4V) hodnota díky které můžeme měnit amplitudu na kanálu 1
double	frekvencech1	(V HZ!, MAX 100Hz) hodnota díky které můžeme měnit frekvenci na kanálu 1
double	posunch2	(NAPETI!, MAX 4V) hodnota díky které můžeme posunout začátek signálu kanálu 2
double	amplitudach2	(NAPETI!, MAX 4V) hodnota díky které můžeme měnit amplitudu na kanálu 2
double	frekvencech2	(V HZ!, MAX 100Hz) hodnota díky které můžeme měnit frekvenci na kanálu 2

Vývojový diagram



Výpis programu

Main.c

```
1.  #include <avr/io.h>
2.  #include <stdio.h>
3.  #include <util/delay.h>
4.  #include <math.h>
5.
6.  //FREKVENCE = 1/T * 1000
7.  //Bod Sinusu = Amplituda/2 * sin(2PI * frekvence * krok(cas) + posun) + Amplituda/2 (+ amplituda
    jelikoz by to vychazelo v minusu, /2 obe amplitudy jelikoz by to vychazelo 2x nastavene amplitude)
8.  //255 => +- 4,3V
9.  //210 => 4V (Nami urcena maximalni amplituda!)
10. //1 => +- 0,0190V
11.
12. //PROMENE PRO KROKOVANI SINUSOVEHO SIGNALU
13. int kroksinch1; //Krok sinusu na kanale 1
14. int kroksinch2; //Krok sinusu na kanale 2
15. int sinusch2; //Jeden bod sinusoveho signalu na kanale 1
16. int sinusch1; //Jeden bod sinusoveho signalu na kanale 2
17.
18. //POMOCNE PROMENNE
19. double cisloch1; //Cislo ktere zapisujeme na PORTB (kanal 1)
20. double cisloch2; //cislo ktere zapisujeme na PORTD (kanal 2)
21. double maxch1; //max vychylka kde muze PORTB jit tak aby nesel do saturace(4V)
22. double maxch2; //max vychylka kde muze PORTD jit tak aby nesel do saturace(4V)
23. double cekanich1 = 0; //hodnota pro cekani mezi 0 a amplitudou pri obdelnikovym signalu (kanal 1)
24. double cekanich2 = 0; //hodnota pro cekani mezi 0 a amplitudou pri obdelnikovym signalu (kanal 2)
25.
26. //PROMENNE URCENE PRO PREVOD
27. int bitposunch1; //bitovy posun kanalu 1
28. int bitposunch2; //bitovy posun kanalu 2
29. int amplitudabitch1; //amplituda ch1 v bitovym cisle
30. int amplitudabitch2; //amplituda ch2 v bitovym cisle
31. double frekvencemsch1; //Frekvence ch1 v ms
32. double frekvencemsch2; //Frekvence ch2 v ms
33. double p1; //Promenna pro krok trojuhelniku a pily na kanalu 1
34. double p2; //Promenna pro krok trojuhelniku a pily na kanalu 2
35.
36.
37. //NASTAVENI POSUNU, AMPLITUDY A FREKVENCE
38. double posunch1 = 0; //(NAPETI!, MAX 4V) hodnota díky které můžeme posunout začátek signálu kanálu 1
39. double amplitudach1 = 1; //(NAPETI!, MAX 4V) hodnota díky které můžeme měnit amplitudu na kanálu 1
40. double frekvencech1 = 10; //(V HZ!, MAX 100Hz) hodnota díky které můžeme měnit frekvenci na kanálu 1
41.
42. double posunch2 = 0; //(NAPETI!, MAX 4V) hodnota díky které můžeme posunout začátek signálu kanálu 2
43. double amplitudach2 = 2; //(NAPETI!, MAX 4V) hodnota díky které můžeme měnit amplitudu na kanálu 2
44. double frekvencech2 = 20; //(V HZ!, MAX 100Hz) hodnota díky které můžeme měnit frekvenci na kanálu 2
45.
46. void nastaveniportu()
47. {
48.     DDRB = 0xFF; //Kanal 1, vystup (aby jsme mohli zapisovat)
49.     DDRD = 0xFF; //Kanal 2, vystup (aby jsme mohli zapisovat)
50. }
51.
52. static void nastavenifrekvence() //Prevedeni frekvence z hz na milisekundy + zkontrolovani maximalni
    hodnoty
53. {
54.     if (frekvencech1 > 100) //Kdyz je nastavena vetsi frekvence nez maximum automaticky se nastavi na
    maximum (100Hz)
55.     {
56.         frekvencech1 = 100;
57.     }
58.     else if (frekvencech2 > 100)
59.     {
60.         frekvencech2 = 100;
61.     }
62.     frekvencemsch1 = 1 / frekvencech1 * 500; //*500 Protoze chceme 1/2 cyklu
63.     frekvencemsch2 = 1 / frekvencech2 * 500; //*500 Protoze chceme 1/2 cyklu
```

```
64. }
65.
66. static void nastaveniamplitudy() //Prevedeni amplitudy z napeti na bitove slovo + zkontrolovani
    maximalni hodnoty
67. {
68.     amplitudabitch1 = (int) (amplitudach1 / 0.019); // #include <math.h> then round(cislo)
69.     amplitudabitch2 = (int) (amplitudach2 / 0.019); // int amplitudabitch2 = (int) amplitudach2
70.
71.     if (amplitudabitch1 > 210) //Kdyz je nastavena vetsi amplituda nez maximum (210/4V) tak se
    automaticky nastavi na 4V
72.     {
73.         amplitudabitch1 = 210;
74.     }
75.     if (amplitudabitch2 > 210)
76.     {
77.         amplitudabitch2 = 210;
78.     }
79.
80.     maxch1 = amplitudabitch1;
81.     maxch2 = amplitudabitch2;
82. }
83.
84. static void nastaveniposunu() //Prevedeni posunu z napětí na bitové slovo + zkontrolování maximální
    hodnoty
85. {
86.     bitposunch1 = (int)(posunch1 / 0.019);
87.     bitposunch2 = (int)(posunch2 / 0.019);
88.     if (bitposunch1 > 210) //Kdyz je nastaveny posun vetsi nez maximum tak se automaticky nastavi
    maximum (210/4V)
89.     {
90.         bitposunch1 = 210;
91.     }
92.     if (bitposunch2 > 210)
93.     {
94.         bitposunch2 = 210;
95.     }
96.     cisloch1 = bitposunch1; //nastaveni bitoveho posunu na cisloch1 (tak aby tam signal na kanale 1
    zacinal)
97.     cisloch2 = bitposunch2; //nastaveni bitoveho posunu na cisloch2 (tak aby tam signal na kanale 2
    zacinal)
98. }
99.
100.        static void nastavenikroku()
101.        {
102.            p1 = (amplitudabitch1 / frekvencemsch1) / 2; //Krokovani troj. signalu a pily (/2 aby bylo
    vice kroku pro vetsi frekvence)
103.            p2 = (amplitudabitch2 / frekvencemsch2) / 2; //Krokovani troj. signalu a pily (/2 aby bylo
    vice kroku pro vetsi frekvence)
104.        }
105.
106.        static void zapsaniobdel(int kanal)
107.        {
108.            switch (kanal)
109.                {
110.                    case 1: //kanal 1
111.                        if (cisloch1 > 0 && cisloch1 < amplitudabitch1)
112.                            cisloch1 = 0;
113.                        if (cekanich1 >= frekvencemsch1 && cisloch1 == amplitudabitch1)
114.                        { //kdyz je obdelnik. signal v "1" a pockal po dobu frekvence tak se zresetuje
115.                            cisloch1 = 0;
116.                            cekanich1 = 0;
117.                        }
118.                        else if (cekanich1 >= frekvencemsch1) //kdyz je obdelnik. signal v "0" a pockal
    po dobu frekvence tak se da do "1"
119.                        {
120.                            cisloch1 = amplitudabitch1;
121.                            cekanich1 = 0;
122.                        }
123.                        else
124.                        {
125.                            PORTB = cisloch1;
```

```
126.                             cekanich1++;
127.
128.                         }
129.                         break;
130.                     case 2: //kanal 2
131.                         if (cisloch2 > 0 && cisloch2 < amplitudabitch2)
132.                             cisloch2 = 0;
133.                         if (cekanich2 == frekvencemsch2 && cisloch2 == amplitudabitch2)
134.                         { //kdyz je obdelnik. signal v "1" a pockal po dobu frekvence tak se zresetuje
135.                             cisloch2 = 0;
136.                             cekanich2 = 0;
137.                         }
138.                         else if (cekanich2 == frekvencemsch2) //kdyz je obdelnik. signal v "0" a pockal
    po dobu frekvence tak se da do "1"
139.                         {
140.                             cisloch2 = amplitudabitch2;
141.                             cekanich2 = 0;
142.                         }
143.                         else
144.                         {
145.                             PORTD = cisloch2;
146.                             cekanich2++;
147.                         }
148.                         break;
149.                 }
150.         }
151.
152.         static void zapsanitroj(int kanal)
153.         {
154.             switch (kanal)
155.                 {
156.                     case 1: //kanal 1
157.
158.                         if (cisloch1 >= amplitudabitch1) //kdyz trojuhelnik je v max. hodnote amplitudy
    tak maxch1 pojede dolu a zapisujeme maxch1
159.                         {
160.                             PORTB = maxch1;
161.                             maxch1 = maxch1 - p1;
162.                             if (maxch1 <= 0) //kdyz je maxch1 v nule a cisloch2 maximum tak se zresetuje
163.                             {
164.                                 maxch1 = amplitudabitch1;
165.                                 cisloch1 = 0;
166.                                 PORTB = cisloch1;
167.                             }
168.                         }
169.                         else
170.                         {
171.                             PORTB = cisloch1;
172.                             cisloch1 = cisloch1 + p1;
173.                         }
174.                         break;
175.                     case 2: //kanal 2
176.                         if (cisloch2 >= amplitudabitch2) //kdyz trojuhelnik je v max. hodnote amplitudy
    tak maxch2 pojede dolu a zapisujeme maxch2
177.                         {
178.                             PORTD = maxch2;
179.                             maxch2 = maxch2 - p2;
180.                             if (maxch2 <= 0) //kdyz je maxch2 v nule a cisloch2 maximum tak se zresetuje
181.                             {
182.                                 maxch2 = amplitudabitch2;
183.                                 cisloch2 = 0;
184.                                 PORTD = cisloch2;
185.                             }
186.                         }
187.                         else
188.                         {
189.                             PORTD = cisloch2;
190.                             cisloch2 = cisloch2 + p2;
191.                         }
192.                         break;
193.                 }
```

```
194.        }
195.
196.        static void zapsanipila(int kanal)
197.        {
198.            switch (kanal)
199.                {
200.                    case 1: //kanal 1
201.
202.                        PORTB = cisloch1;
203.                        cisloch1 = cisloch1 + p1/2; //Krok je /2 jelikoz signal pily je 2x kratsi
204.                        if (cisloch1 >= amplitudabitch1)
205.                        {//kdyz je cisloch1 v max. amplitude zmeni se na 0
206.                            cisloch1 = 0;
207.                        }
208.                        break;
209.                    case 2: //kanal 2
210.                        PORTD = cisloch2;
211.                        cisloch2 = cisloch2 + p2/2; //Krok je /2 jelikoz signal pily je 2x kratsi
212.                        if (cisloch2 >= amplitudabitch2)
213.                        { //kdyz je cisloch2 v max. amplitude zmeni se na 0
214.                            cisloch2 = 0;
215.                        }
216.                        break;
217.                }
218.        }
219.
220.        static void zapsanisin(int kanal)
221.        {
222.            switch(kanal)
223.                {
224.                    case 1: //kanal 1
225.                        kroksinch1++; //krok
226.                        sinusch1 = (int) (amplitudabitch1/2 * sin(6.28 * frekvencech1 * kroksinch1 +
    posunch1) + amplitudabitch1/2);
227.                        if (kroksinch1 == frekvencemsch1*4)
228.                            kroksinch1 = 0; //po 4 vykresleni sinusovky se krok vynuluje aby se krok
    nebyl moc velký
229.                        PORTB = sinusch1;
230.                        break;
231.
232.                    case 2: //kanal 2
233.                        kroksinch2++; //krok
234.                        sinusch2 = (int) (amplitudabitch2/2 * sin(6.28 * frekvencech2 * kroksinch2 +
    posunch2) + amplitudabitch2/2);
235.                        if (kroksinch1 == frekvencemsch1*4)
236.                            kroksinch1 = 0; //po 4 vykresleni sinusovky se krok vynuluje aby se krok
    nebyl moc velký
237.                        PORTD = sinusch2;
238.                        break;
239.                }
240.         }
241.
242.         int main (void) {
243.            nastaveniportu();
244.            nastavenifrekvence();
245.            nastaveniamplitudy();
246.            nastaveniposunu();
247.            nastavenikroku();
248.            while (1) {
249.                _delay_ms(1); //Delay pred kazdym krokem 1ms
250.
251.                //KANAL 1
252.
253.                //zapsaniobdel(1);
254.                //zapsanitroj(1);
255.                //zapsanipila(1);
256.                zapsanisin(1);
257.
258.                //KANAL 2
259.
260.                zapsaniobdel(2);
```

```
261.                //zapsanitroj(2);
262.                //zapsanipila(2);
263.                //zapsanisin(2);
264.            }
265.        }
```

Odpovědi na otázky:

1. Funkční generátor může umožňovat modulovat základní signál jednoho kanálu aktuální úrovní signálu z kanálu druhého. Zkuste naznačit jak by bylo možné implementovat AM modulaci na kanálu 1 aktuální úrovní signálu na kanálu 2 v rámci vašeho řešení

2. Často se můžete setkat s jednobitovými převodníky využívající PWM. Pokuste se odvodit jaká by byla maximální frekvence reprodukovatelného signálu pro PWM DAC s frekvencí 16MHz při použití rozlišení 8 a 12bitů (tip: rozlišení určuje počet tiků základní frekvence, použití Nyquistova teorému).

3. V rámci integrovaných převodníků se často používá sériové rozhraní SPI. Popište jak vypadá přenos informace tímto rozhraním a uveďte příklad jednoho konkrétního DAC s tímto rozhraním.

Závěr:

V úloze jsme špatně vymysleli převodník frekvence a amplitudy, nebyla ani trochu přesná, šlo ale vykreslovat na oba kanály zvlášť jiný signál. Chybělo seriové rozhraní aby šlo nastavovat všechno přes PuTTy, takže se všechno nastavovalo skrz kód, ale zadání je tak nějak splněno.