Pokročilé programovanie mikropočítačov v jazyku C. Diel č.1

Publikované: 02.10.2015, Kategória: Mikroprocesory

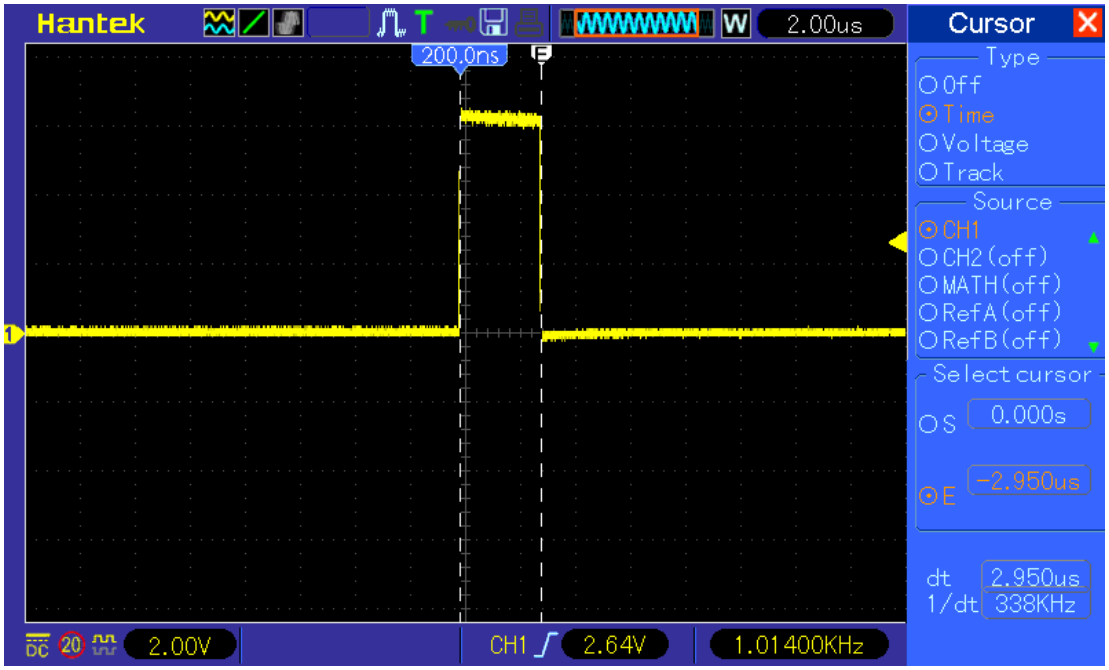
www.svetelektro.com

Float, int alebo char?

Ahojte programátori. Som na tomto fóre zaregistrovaný už pár rokov a mojou oblúbenou diskusnou témou sú mikroprocesory. Každým rokom sa tu objaví množstvo nových programátorov, ktorý hľadajú rady a odpovede na jednoduché ale aj zložitejšie otázky ohladne programovania rôznych typov mikropočitačov. Zawinova knižka Programujeme AVR v jazyku C je veľmi užitočným návodom pre začínajúcich programátorov. Ja som si pripravil sériu článkov, ktoré sa venujú ani nie tak pokročilím metódam programovania ale skôr poukazujú na záludnosti alebo krásu programovania mikroprocesorov v jazyku C.

Prvý článok je zameraný na správny výber dátového typu premmenej. Premenné sú základom každého programu. V malých domácich projektoch na výbere správneho dátového typu nemusí až tak veľmi záležať ale pri velkých projektoch sa každý ušetrený byte počíta. Takisto nejde len o šetrenie miesta ale aj o dobu trvania tej ktorej operácie s daným dátovým typom, čo pri niektoržch časovo kritických slučkách hrá dôležitú úlohu.
 Pomocou jednoduchého programu a osciloskopu porovnávam dĺžku trvania výpočtu jednoduchých matematických operácii. Konkrétnejšie, mám dve premenné A a B. V prvom prépade sú dátového typu float. Premenné A a B inicializujem ľubovolným desatinným číslom. Následne vykonám s premennými A a B základne matimatické operácie, sčítanie, odčítanie, násobenie a delenie. Pomocou osciloskopu si odmeriam čas trvania matematickej operácie a to tak, že tesne pred samotnou operáciou nahodím ľubovolný nevyužitý pin mikroprocesora do logickej jednotky a okamžite po skončení operácie ho zase zhodím do logickej nuly. Túto zmenu stavu pinu si odsledujem na osciloskope.
Jednoduchý obslužný program na sledovanie doby operácie:

```
#include <avr/io.h>
   #include <util/delay.h>

      int a = 2;
      int b = 3;
      int c;

   int main(){

      DDRD = 0x01;         // PD0 ako vystupný pin ostatné ako vstupne
      PORTD= 0x00;         // PD0 do nuly
      // nekoneèná sluèka
      while(1){
         PORTD |= (1 << PD0); // PD0 do jednotky

                  c = a*b1;

                  PORTD &= ~(1 << PD0); // PD0 do nuly

                  _delay_ms(1);
      }

   return 0;
   }
```



*Doba trvania – celočíselné násobenie int * int*

Pre dátové typy char, int a long použijem rovnaký program len premenným A a B zmením datový typ. Tento pokus robím na dobre známom mikropočítači Atmega8. Interný oscilátor mi beží na 8Mhz. Namerané časy jednotlivých matematických operácii su zaznamenané v tabuľke.

Tab.1 Doby trvania matematických operácii s rôznymi dátovými typmi

dátový typ	**float**		**char**		**int**		**long**	
matematická operácia	**čas [us]**	**počet cyklov**	**čas [us]**	**počet cyklov**	**čas [us]**	**počet cyklov**	**čas [us]**	**počet cyklov**
+	100	800	1.15	9	2	16	3.6	28
-	110	880	1.15	9	2	16	3.6	28
*****	240	1920	1.5	12	2.9	23	9.1	72
/	149	1192	10.7	85.6	30	240	78.5	628
>>1	n/a		0.9	7	1.5	12	2.6	20
<<1			0.9	7	1.5	12	2.6	20
<<2			0.9	7	1.7	13	3.9	31

Premenná typu char je 8 bitová celočíselná, int je 16 bitový a long by mal byť 32 bitový. Osem bitové operácie vykonáva Atmega 8 celkom svižne až na operáciu celočíselného delenia. V prípade ak delíme číslom 2,4,8.. je ovela lepšie použiť bitový posun ako operáciu delenia. Čisto teoreticky správny compiler by si to mohol ošetriť aj sám ale v mojom prípade to tak nebolo. Podľa očakávaní operácie s dátovým typom float trvajú najdlhšie. Dovolím si povedať, že veľmi dlho. Preto je veľmi výhodne sa dátovému typu float skôr vyhýbať ako ho využívať. Osobne ma prekvapilo, že operácia delenia trvá kratšie ako operácia násobenia pri floate. Je to zrejme featura atmieg.

Možno si poviete. Dobre to som vedel aj predtým že matematické operácie s float premennými trvajú najdlhšie. A keď budem potrebovať zrýchliť výpočet tak použijem modernejší mikrokontrolér s FPU jednotkou a nie Atmegu8. Tak trochu súhlasim. No veľmi dolezitú úlohu v tabuľke hra aj stĺpček, ktorý popisuje počet cyklov potrebný pre danú operáciu. Počet cyklov je vypočítaný podľa veľmi jednoduchého vzorca.
Počet cyklov = čas trvania operácie x frekvencia jadra (u mňa 8 Mhz)
Pre atmegu8 a pre malé domáce projekty to moc nehrá veľkú úlohu, ale v prípade že je dôležité brať ohlad aj na spotrebu čipu v danej aplikacií je dobre vedieť či sa mi náhodou nevyplatí znížiť frekvenciu jadra a šetriť energiu batérie (pozri tabulku na strana 292 z datasheetu Atmegy 8).
Toľko prvý článok. Teším sa na ďalšie pokračovanie a na vaše postrehy k tomuto článku.
Prajem štastnú ruku pri výbere dátových typov premenných.