PBM op AVR

Willem Ouwerkerk

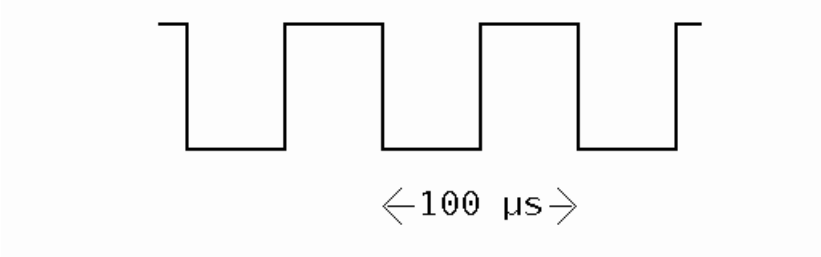
In deze tekst worden drie methodes beschreven voor PulsBreedte Modulatie op de AVR. Je kunt dit artikel, inclusief de volledige ByteForth sources, downloaden als ZIP-file. Zie onderaan deze pagina.

Inleiding

PBM of langer gezegd PulsBreedteModulatie is een methode om op een digitale manier vermogen te regelen of data te versturen. Bijvoorbeeld de helderheid van een lamp, de snelheid van een motor, de temperatuur van een verwarmingselement, de stand van een servomotor, DAC (Digitaal Analoog Convertor = het omgekeerde van ADC), etc. Het is niet de eerste keer dat we PBM behandelen, maar de AVR-hardware en de getoonde software oplossing rechtvaardigen het. Zie daarom ook Vijgeblaadje nr. 4 en nr. 19.

PBM wordt over het algemeen opgewekt door een blokgolf met vaste frequentie te gebruiken. De tijdsduur van het hoge en lage deel wordt gevarieerd. Er wordt gelijkspanning gebruikt waar bijvoorbeeld 10000 keer per seconde een stuk uit wordt geknipt. Naarmate dat stuk groter is neemt het vermogen af. Bij een frequentie van 10 KHz is de herhalingstijd:

```
t/f = 1/10000 = 100 µs
```



Een van de grote voordelen van deze methode is, dat er er weinig hardware nodig is om vermogen te regelen. Vaak is een enkele MOSFET-transistor voldoende, ook gaat er zeer weinig vermogen verloren en is er per uitgang maar een ordinair I/O-bit nodig.

Hoe wekken we PBM op?

We kunnen PBM met software of speciale hardware opwekken. De softwaremethode heeft als voordeel dat we niet afhankelijk zijn van de aanwezigheid van speciale hardware, dat we zelf een I/O-pin kunnen kiezen en dat het aantal PBM-uitgangen zelf kunnen bepalen. Het nadeel is dat software PBM een nogal zware belasting op de CPU legt, die zwaarder wordt bij een hogere PBM frequentie.
Hardware PBM heeft deze nadelen niet, maar je bent hier weer afhankelijk van het aantal PBM-modulatoren dat de ontwerpers van de chip eraan toe hebben gevoegd. Meestal is hardwarematig vastgelegd op welke I/O-bits de uitgangen aangesloten zijn.

Software PBM

Voordat we beginnen met het coderen van de software moeten we nauwkeurig vaststellen waar de PBM aan moet voldoen:

- De gewenste herhalingsfrequentie
- Het regelbereik
- Aantal PBM uitgangen

Voorbeeld 1 (software methode)

Frequentie ~10 KHz, regelbereik 2% tot 98% in 50 stappen en een PBM uitgang op poort-B en pen-0. De gebruikte AVR is de AT90S2313.
MMhhh, simpel klusje....

We kiezen timer-0, die heeft een resolutie van 8-bits. De hier gebruikte softwaremethode belast de CPU wat minder waardoor we gemakkelijk een hogere frequentie kunnen gebruiken. Door vooraf de tijdsduur van de hoge en lage periode te berekenen hoeft de interrupt routine hoeft slechts twee maal per periode aangeroepen te worden. Bij meer gangbare implementaties wordt de interrupt net zo vaak aangeroepen als het aantal stappen van het regelbereik, hier 50 en dat scheelt nogal. We gaan er vanuit dat het gebruikte kristal er een van 4 MHz is. De namen PORTB, TCCR0, TCNT0, etc. zijn labels van hardware adressen op de AVR-controller. AVR-ByteForth gebruikt hiervoor dezelfde namen als de fabrikant.

```
PORTB 0 BIT-SFR UITGANG  \ Gewenste PBM uitgangsbit
TCCR0 SFR TIMER-CONTROLE \ Timer besturing
TCNT0 SFR TIMER          \ Timer register
TIMSK SFR TIMER-INTRPT   \ Timer interrupt activering

#50 CONSTANT CYCLUS      \ Max. periodeduur
REGISTER AANPULS         \ Tijd dat uitgang hoog is
REGISTER UITTIJD         \ Tijd dat uitgang laag is

CODE PBM ( -- )
  ADR UITGANG SBIS, \ Sla AHEAD, over als uitgang hoog is
  AHEAD,                        \ Ga door na THEN,
     ADR TIMER ADR UITTIJD OUT, \ Zet duur lage periode
     ADR UITGANG CBI,           \ Uitgang laag
     RETI,
  THEN,
  ADR TIMER ADR AANTIJD OUT,    \ Zet duur hoge periode
  ADR UITGANG SBI,              \ Uitgang hoog
  RETI,
END-CODE  T0-OVERFLOW

: SNELHEID ( +n -- )
  1 UMAX CYCLUS 1- UMIN  \ Houdt +n tussen 1 en cyclus-1
  DUP NEGATE TO AANTIJD  \ Zet aan periode
  CYCLUS - TO UITTIJD    \ Zet uit periode
  ;

: SETUP-PBM  ( -- )
  -1 TO TIMER            \ Dummy timer inhoud bij opstarten
  2 TO TIMER-INTRPT      \ timer-0 interrupt aan
  2 TO TIMER-CONTROLE    \ prescaler=8
  1 SETDIR UITGANG       \ PB.0 is uitgang
  ;
```

De interrupt routine gebruikt ongeveer 8% van de beschikbare processortijd, dat is voor een PBM-frequentie van 10 KHz erg weinig.

Wedstrijd

In dit voorbeeld kan het vermogen niet van 0% tot 100% geregeld worden. Bedenk zelf een uitbreiding die dit wel mogelijk maakt.

Het meest originele antwoord is destijds beloond met een AVR-ByteForth inclusief programmeerdongle. De oplossingen in de vorm van een volledig en uitvoerbaar programma konden tot 1 juni 2003 ingstuurd worden. Bestuursleden van de HCC Forth-gg mochten alleen buiten mededingen meedoen.

Winnaar werd Gerard van der Sel.

Voorbeeld 2 (hardware methode)

De moderne AVR-microcontrollers hebben vaak een of meer hardware PBM uitgangen. We nemen weer een hoge uitgangsfrequentie van ~10 KHz de I/O-pin ligt vast in de AVR-hardware, het regelbereik ook. De uitgang is OC1 oftewel poort-B pen-3. Ook hier gaan we er vanuit dat het gebruikte kristal voor de AT90S2313 er een van 4 MHz is.

```
PORTB 3 BIT-SFR UITGANG \ Door hardware opgelegde PBMuitgang

TCCR1A SFR CONTROLE-A     \ Stel PBM-type in
TCCR1B SFR CONTROLE-B     \ Stel klok ingang in
OCR1AL SFR SNELHEID       \ 0 = uit, 255 is max. 100%

: SETUP-PBM  ( -- )
  $81 TO CONTROLE-A       \ Normale 8-bit PBM mode
  $01 TO CONTROLE-B       \ CPU-klok als ingangsfreq.
  1 SETDIR UITGANG        \ PB.3 is uitgang
  0 TO SNELHEID           \ PBM uitgang op 0 %
  ;
```

Het grote voordeel van de hardware methode is dat het opwekken van PBM met een maximumfrequentie van 7843 Hz (bij een 4 MHz kristal) geen cpu tijd kost. We hoeven slechts het hardware register SNELHEID te voorzien van een getal, de rest doet de AVR-hardware. Ook het regelbereik is groter, 0 tot 255 i.p.v. 0 tot 50 zonder dat er extra CPU-tijd in gaat zitten. Wat moet je doen om in dit tweede voorbeeld ook een bereik van 0 tot 50 te krijgen?

Slot

PBM is een zeer nuttig mechanisme. Er zijn veel verschillende manieren te bedenken om het zinvol in te zetten. De AVR kan door zijn ingebouwde hardware en efficiente software hoge PBM-frequenties genereren. De processor houdt hierbij nog tijd genoeg over om andere taken te vervullen. Als de AVR op een hogere kristalfrequentie geklokt wordt komen die cijfers nog gunstiger uit. De gekozen AT90S2313 kan op maximaal 10 MHz geklokt worden. Modernere AVR's als de ATtiny26 kunnen zelfs op 16 MHz geklokt worden hierdoor worden alle cijfers nog vele malen gunstiger.

download

PBM artikel en sources