Chauffe-capa

v 2.2

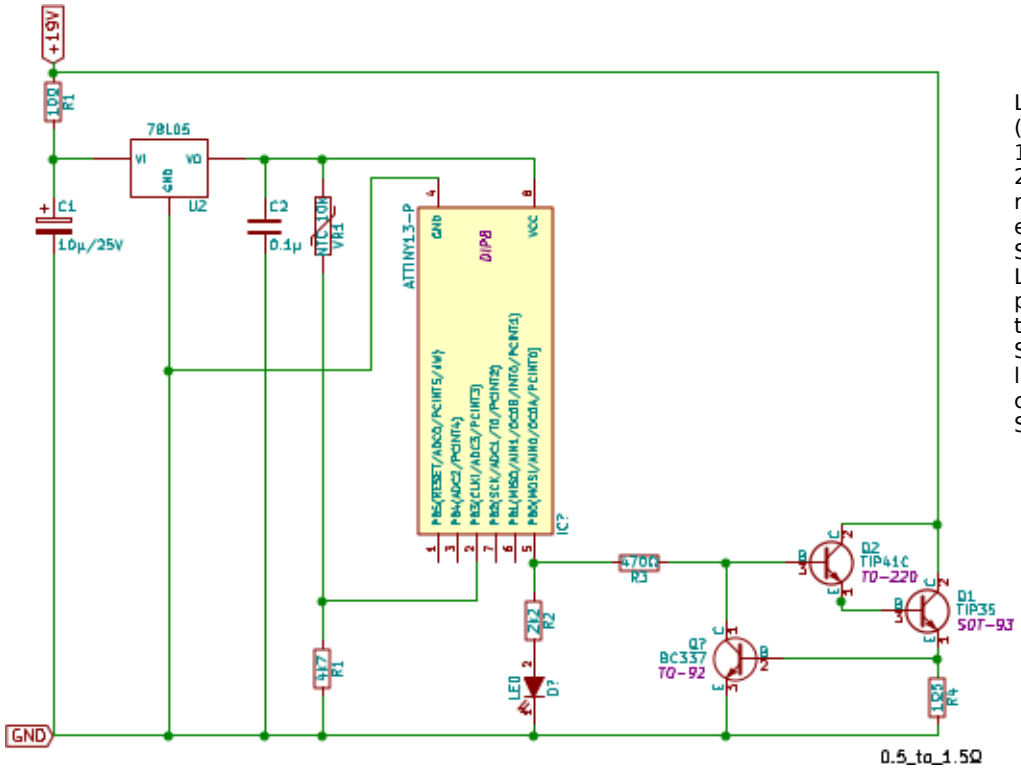
Les condensateurs d'allumage sont sensibles à la température, c'est d'ailleurs un excellent moyen d'accentuer ou de révéler les défauts. Ce qui était d'ailleurs utilisé par le Bermascope. Le Spithascope en possède donc un ! Mais plus précis : celui du Bermascope n'avait pas de limitation, juste l'asymptote de chauffage.
Un clip métallique maintient le condensateur et lui transmet plus ou moins bien la chaleur.
Le chauffage est fait ici par la dissipation dans un transistor, avec limitation du courant.

Architecture

- un microcontrôleur Atmel AT tiny13 (à 0,2$) qui mesure et contrôle la température, horloge interne, programmé
- une thermistance 10k en tube inox (à 0,06$)
- un transistor de chauffage TIP41C (couplage thermique serré et inamovible avec la thermistance)
- un transistor de limitation BC547
- une résistance de mesure de courant de 0,5Ω à 1,5Ω ±5 % 1W, par exemple deux résistances de 2,4Ω 0,5W en parallèle
- le tout alimenté par le bloc principal : alim PC portable 19V 3,5A

La chaufferette

C'est un transistor qui chauffe, plongé dans le bocal où on trempe la capa à chauffer.



Le TIP35 peut dissiper 70W @80°C (derating) et sa SOA auiorise 19V/8A, on lui en fait passer 10 à 20W. il lui faut un minimum de radiateur ou de masse thermique, et l'isoler de la face avant du Spithascope.
Le tiny13 fournit un signal 0-5V pulsé PWM pour le contrôle de la température.
Si on applique du 5V continu à l'étage de sortie, le chauffe-capa chauffe à donf (>100°C),
Si on met 0V, il s'éteint.

Le montage limite le courant à 0,5A jusqu'à 1,3A. Sur l'alim de 19V, la dissipation max dans le transistor est de 10W @0,5Aet la température atteint largement les 100°C sur radiateur 10°/W sans paraffine. La régulation par µC et thermistance va chauffer la capa à tester vers 70°C, jugée moyennement dangereuse, l'utilisateur devra être averti de cette température et prendre ses précautions, la température des chauffes-eau est d'ailleurs limitée légalement à ces températures. Comme les condensateurs à tester ont des masses et des caractéristiques thermiques très différentes, il n'est pas possible de faire une régulation fine elle sera de type thermostatique avec plage proportionnelle et intégrale.
Selon le volume de paraffine à chauffer, on peut passer la consigne MAX_TEMP à 830 et la résistance shunt de mesure à 0,5Ω.

www.hackerschicken.eu zibuth27@gmx.fr

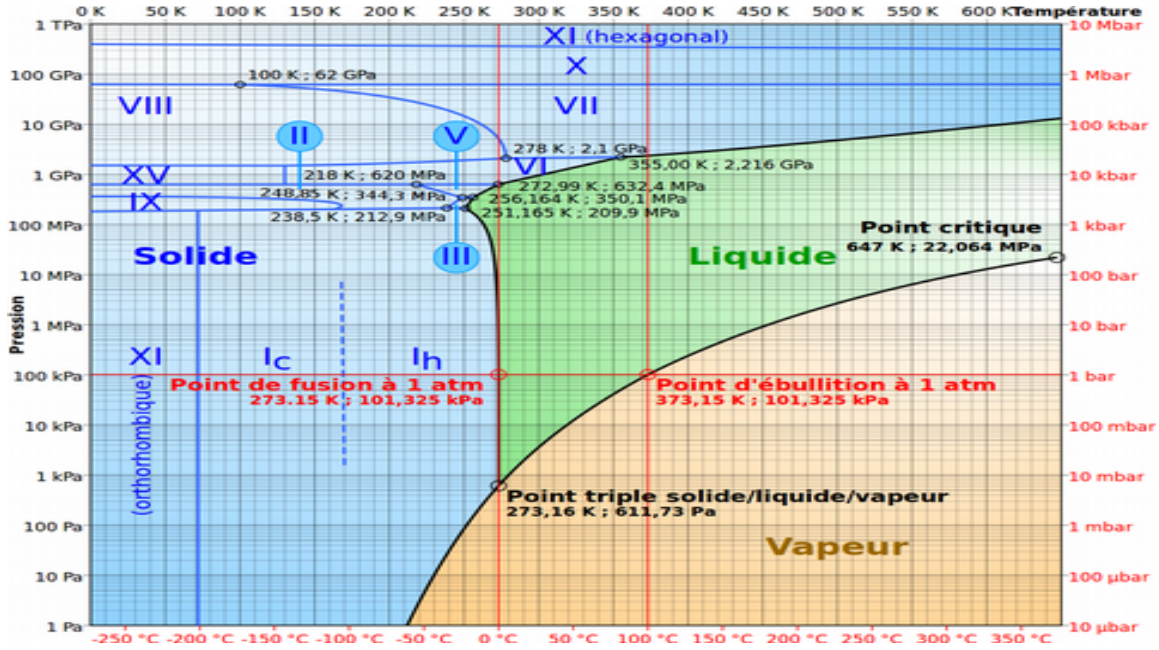
Alimentation
Le montage est alimenté par un 78L05 avec 10 µF/25V en entrée, 47 nF en sortie et 10 Ω en série amont.

Thermistance
Intégrée dans un tube inox. On ne sait rien du remplissage entre la thermistance et le boîtier, donc de la transmission thermique entre le boîtier et la thermistance. Cela affecte surtout la constante de temps de réponse.
Etalonnée en trois points : glace fondante =12189, 23°=10920, 49°=4320

puis extrapolation de la courbe :
http://www.thinksrs.com/downloads/programs/Therm%20Calc/NTCCalibrator/NTCcalculator.htm

La température de la glace fondante est quasiment insensible à la météo ou à l'altitude contrairement à la température d'ébullition, voir le diagramme de l'eau :

source Wikimedia

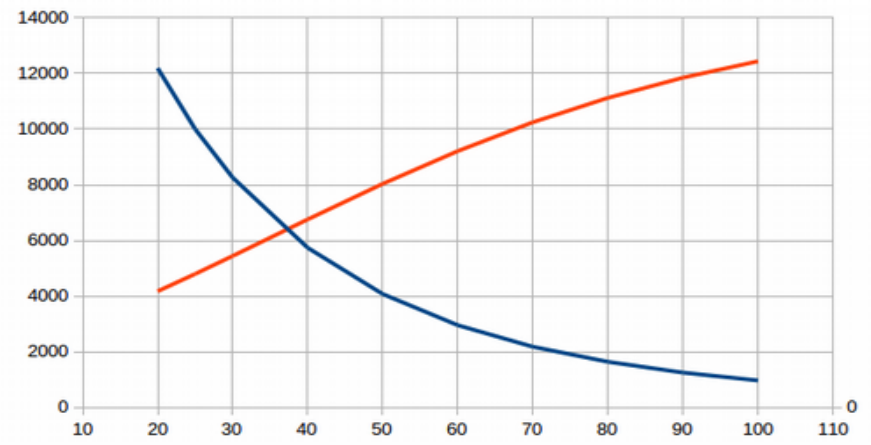


Attention, la température de la glace fondante n'est PAS la température du fond du récipient où on fait fondre la glace. Si on n'agite pas en permanence **l'eau du fond est à +4°C** en raison du comportement anormal de l'eau : la densité de l'eau est maximale à +4°C, c'est donc cette eau à +4°C qui descend au fond !
La présence éventuelle d'une gaine thermorétractable n'est pas du tout la garantie d'une étanchéité parfaite, ça peut n'être qu'un cache-misère, Un bouchon époxy est meilleur, sans être absolu.

On en extrait la table, le β (3590) et la figure suivante (la courbe rouge de tension est multipliée par x, juste pour remplir correctement la figure)

temp	Rth	Vµc
2,9	26200	0,76
25	10000	1,60
30	8257	1,81
40	5735	2,25
50	4073	2,68
60	2954	3,07
70	2182	3,41
80	1640	3,71
90	1252	3,95
100	970	4,14

Ce tableau provient des mesures faites sur un échantillon, les composants installés sont identiques dans la limite des 1 % de la précision vendue.

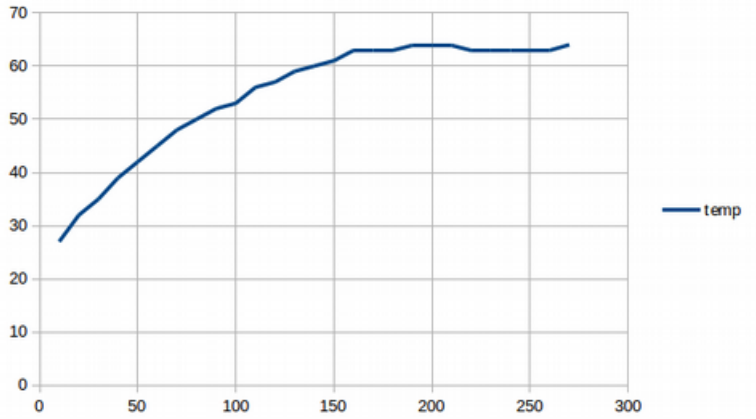


thermistance 10kΩ (à 25°C)
Une résistance série de 4k7 linéarise suffisamment sa réponse en température

La température de 65°C va donner une lecture de la CTN de 3,84V (698 en lecture ADC 10 bits)

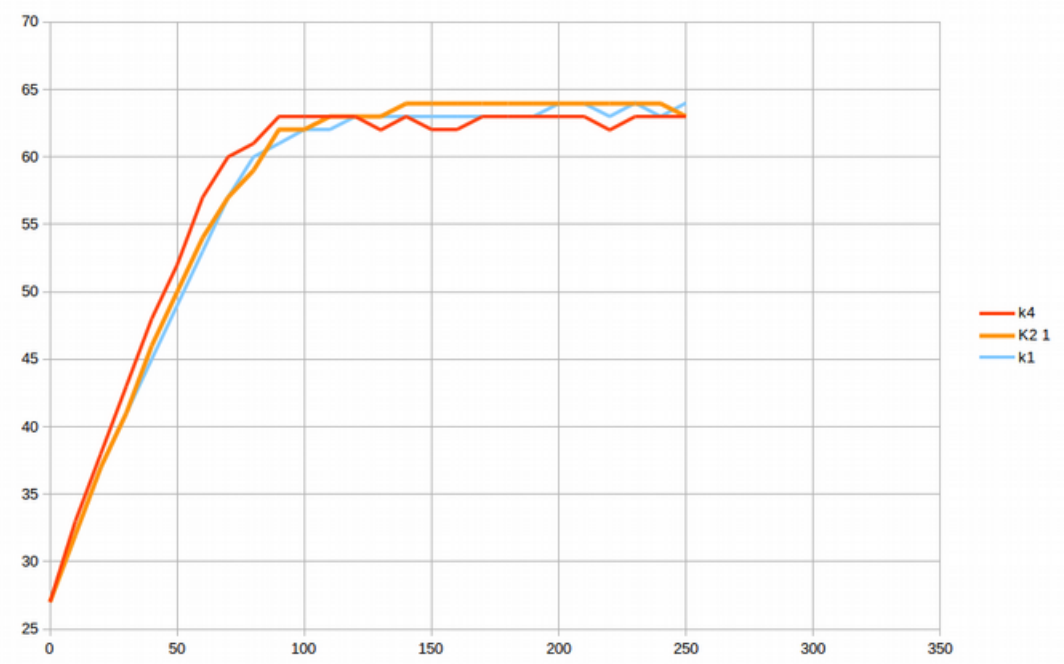
On peut utiliser pratiquement un incrément de 32 pas de PWM. Donc avec un max calé à 70°, la proportionnalité commencera à 60°C. Une LED rouge indique le passage en mode PWM, soit la température quasiment atteinte et l'entrée en zone proportionnelle. Une LED orange indique la mise sous tension. Il n'y a pratiquement pas d'overshoot, il est bien inutile de rajouter d'autres modes de régulation, pour l'usage prévu !

Résultats
Le transistor du chauffe-capa atteint la température finale en moins de 3 minutes. Il faut après, transmettre la chaleur au bloc de paraffine.



Pour réduire l'offset (différence entre la grandeur de consigne et la grandeur finale, on peut aussi parler d'erreur statique, ici égale à 5%), il faut au moins introduire un terme intégral (somme des n dernières erreurs). Cela améliore considérablement l'erreur globale.

Un ki de 4 montre des tendances oscillatoires, en adoptant un ki de 2 avec une consigne à 3,369V on obtient un résultat très correct avec une mise en température en 2 minutes. Mise en température de la source chaude = transistor avec couplage serré à la thermistance. La mise en température du condensateur sera plus longue, et peut perdre plusieurs degrés par rapport au transistor, selon la réalisation mécano-thermique. La différence entre les valeurs du terme intégral sont assez faibles, mais dès qu'il existe, il améliore l'erreur statique, je ne pense pas gagner quelque chose avec le terme différentiel (PID) puisque la vitesse de montée est limitée par les paramètres électriques de puissance choisis. On en reste là !
L'intégration se fait sur cinq relevés d'erreur, soit sur une seconde.
Le résultat est largement au-delà des besoins d'un chauffe-capa, mais il restait de la place en mémoire, j'en ai profité



L'élévation de température finale du condensateur prendra plus de temps, jusqu'à une heure, variable selon la quantité de paraffine. Les « oscillations » ne font que 1°C, c'est à dire qu'il s'agit simplement de discerner l'incrément de lecture du thermocouple du bruit.

On ne devrait pas avoir besoin de réglage (potentiomètre ou modif du code source) pour ce chauffe-capa

Soft :
un antialiasing (anti-repliement de spectre) permet l'utilisation directe de la grandeur capteur en 10 bits et d'utiliser les 8 bits finaux sans se soucier des n fois 8 bits précédant la comparaison proportionnelle. Il permet aussi de faire le cast variable 16b dans le registre OCR0A de 8 bits sans crainte d'altération d'info.

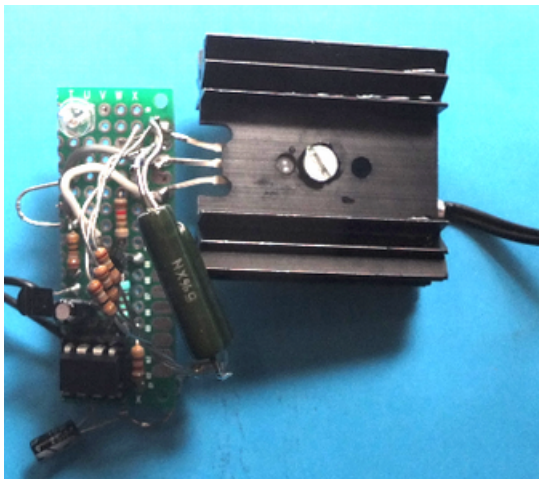
Chauffe-capa

v 2.2

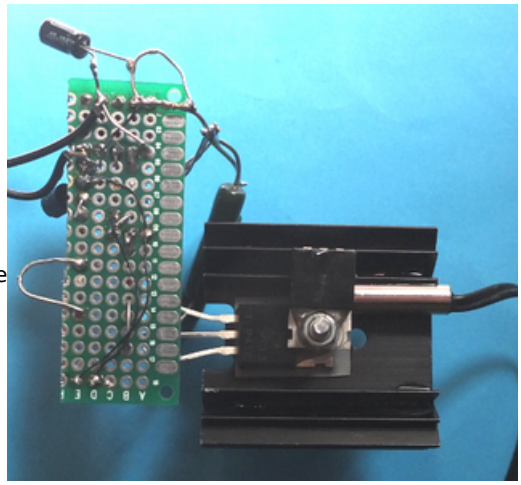
Utilisation

Le chauffe-capa est mis en œuvre par l'alimentation en +19V (commutateur général)

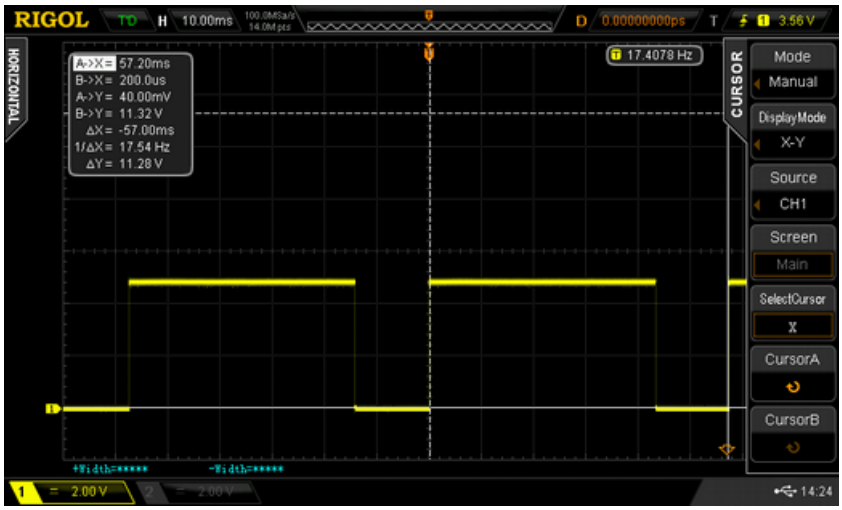
- la mise en chauffe est faite dès l'application du +19V.
- la LED s'allume. C'est la phase de montée en température
- la LED clignote = température atteinte, phase proportionnelle



Couplage serré entre la sonde CTN
et le transistor de chauffage



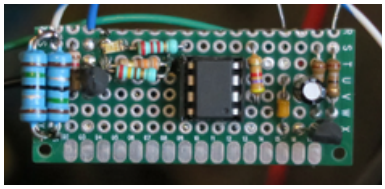
Le proto du chauffe-capa, avant l'installation du clip de fixation

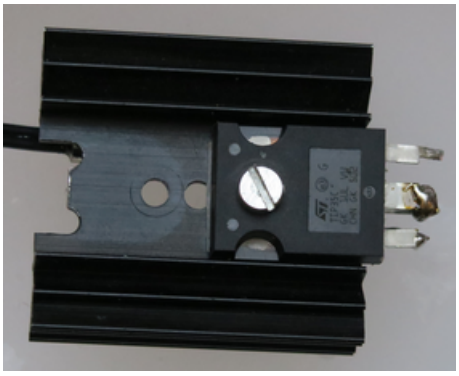


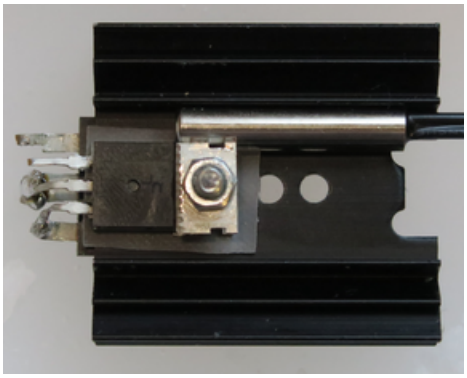
à 65°C le PWM cycle à 75 % de 17Hz

seconde édition
dans un pot plus gros (75mm de diamètre, de quoi tremper largement la capa à tester)

vue du transistor chauffant à travers la paraffine liquide
les fils de connexion traversent le récipient. Ils sont bloqués en place par de l'époxy. Compte tenu de la durée et des cyclages prévus de fonctionnement (l'époxy peut perdre son étanchéité par polymérisation complète / durcissement et cyclages thermiques), j'ai donc rajouté une couche de silicone qui restera élastique, mais tient moins bien les fils de liaison
bien entendu, il y a une rondelle élastique genre onduflex, pour compenser les diffférentiels de dilatation







Le programme :

```
 /* at tiny13
 *
 * ignition capacitor heater
 * proportional control from 60°C to 65°C
 *
 * ports:
 * PB0 pin5 = out PWM
 * PB1 pin6 =
 * PB3 pin2 = ADC3  = thermistor to +5, 4k7 to GND
 * PB2 pin7 =
 * PB4 pin3 = LED : temp Hi
 *
 * proportional from 50 to 70°C
 * full power below 50°
 *
 * Zibuth27 2016/07/30
 * RMZ#228
 * status: OK
 * keywords: ADC, PWM
 * efuse FF hfuse FF lfuse 6A
 */

#include <util/delay.h>
#include <avr/io.h>
#include <avr/sfr_defs.h>
#include <avr/interrupt.h>

#define F_CPU 1200000UL                                 // 1.2 MHz
#define MAX_TEMP 850

volatile uint8_t heat_on = 0;
 int   main(void)
{
//      uint8_t cmd;
        uint16_t temperature, kp, ki;
        uint16_t last_err [5]= {0,0,0,0,0};

        DDRB = 0x11;                                    // output OC0A enable + PB4
        PORTB |= (1<<PB1);                              // pull-up

        TCCR0A = 0x83;                                  // fast PWM mode
        TCCR0B = 0x04;                                  // clk/

        ADMUX  = 0x03 ;                                 // mux0 PC0
        ADCSRA = 0x85;                                  // enable ADC, prescaler 1024

        MCUCR = 0x02;
        GIMSK = 0x40;

//      sei();

  while (1)                                             // endless
  {
//        spst = (PINB & (1<<PB1));
        heat_on = 1;                                    // permanent heat if power on
          if (heat_on) {                                // state polling
                  TCCR0A = 0x83;                        // restart PWM
                  TCCR0B = 0x04;

// temperature capture
                ADCSRA |= (1<<ADSC);
                while (ADCSRA & (1<<ADSC));
                temperature = ADC;

                for (int8_t i =4; i>=1; i--) last_err [i] = last_err [i-1];
                last_err [0] = (MAX_TEMP - temperature);
                ki = (last_err[4] + last_err[3] + last_err[2] + last_err[1] + last_err[4])/5;
                if (ki>254) ki = 254;                   // not 255, to help to check with scope


// antialisasing & heating
                if( temperature < (MAX_TEMP - 100))
                        {
                                OCR0A = 254;            // was 31
                                PORTB &= ~(1<<PB4);     // Hi temp LED off
                        }
```

```
                else {                                                  // PWM mode
                        PORTB |= (1<<PB4);                      // Hi temp LED ON
                        kp = (MAX_TEMP - temperature) ;
                        OCR0A = (10*kp)+(3*ki);           // PID computation       was 8p 2i without wax
                        if (temperature > (MAX_TEMP - 1)) OCR0A = 1;       // not 0, to help to check with scope
                }
                _delay_ms(100);
        }
                        else {                                          // if heat off
                                TCCR0A = 0;
                                TCCR0B = 0;
                                }
  }             // end of while
  return 0;
}

ISR (INT0_vect) {
}
```