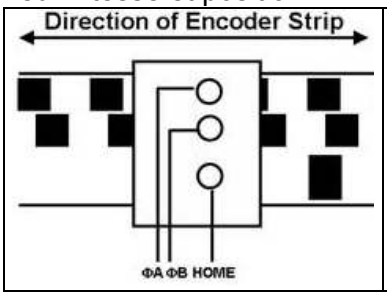
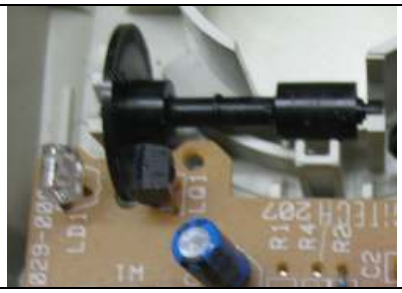
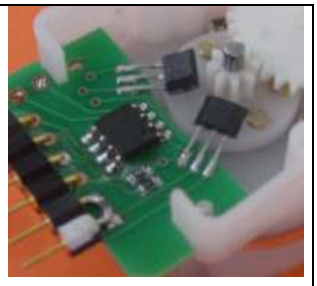
Encodeur

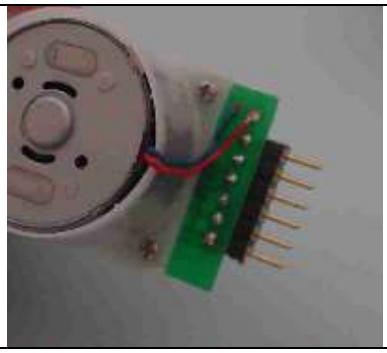
Pour connaître un déplacement linéaire ou rotatif, deux capteurs et une ‘’crémaillère’’ génèrent des signaux déphasés qui sont faciles à décoder. Les souris mécaniques utilisent de tels encodeurs, dits codeurs incrémentaux (quadrature encoders). Les potentiomètres sont de plus en plus remplacés par des encodeurs. Les moteurs continus ont besoin d’un encodeur pour asservir leur vitesse et position.







Le moteur Rome et son encodeur RomEnco de Didel et Solarbotics www.didel.com/mot/RomEnco.pdf convient bien pour expérimenter avec un microcontrôleur. Le MiniRomEnco est en développement.



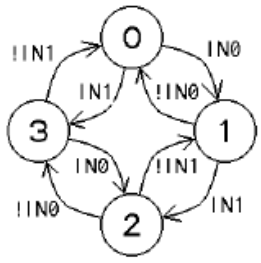
L’analyse de la séquence se fait avec une machine d’état. La structure C switch-case permet de décider dans chaque état quelle est l’évolution future. Dans l’état 0 par exemple, il y a 3 possibilités car on ne peut pas avoir IN0 et IN1 qui passent ensemble à 1 :
IN0 = 1 on passe à l’état 1 et on compte
IN1 = 1 on passe dans l’état 3 et on décompte
IN0 = 0 et IN1 = 0 on reste dans l’état 0

Si les signaux ont des rebonds de contact, comme pour un encodeur mécanique, l’échantillonnage doit se faire tous les 5 à 10ms,

Si le moteur Rome tourne à 5V à vide, les transitions se suivent à 0.4 ms ou plus. Il faut donc échantillonner au moins toutes les 300 microsecondes. La durée de l’échantillonnage est de 5 microsecondes (donc 60 instructions). Cette durée est importante à connaître ; elle dit que la fréquence maximale des signaux d’un encodeur de moteur industriel que l’on peut décoder est de 50 kHz, en occupant compètement notre processeur à 48 MHz.
En programmant avec l’algorithme de Sommer décrit sous www.didel.com/diduino/EncodeurMin.pdf est 5 fois plus rapide (voir plus loin).

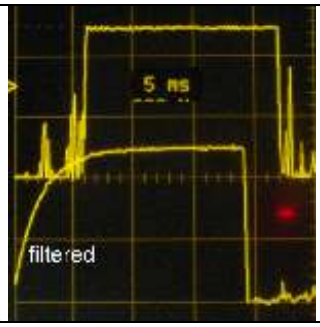
C:\JDN\0Pinguino\KiMotEnco\Enco01.pde

```
// Enco01.pde  enco--> compte/décompte portB
/*
   0 1 2 3 0 1 3 1 0 3 2 1 0
e1
e2
```



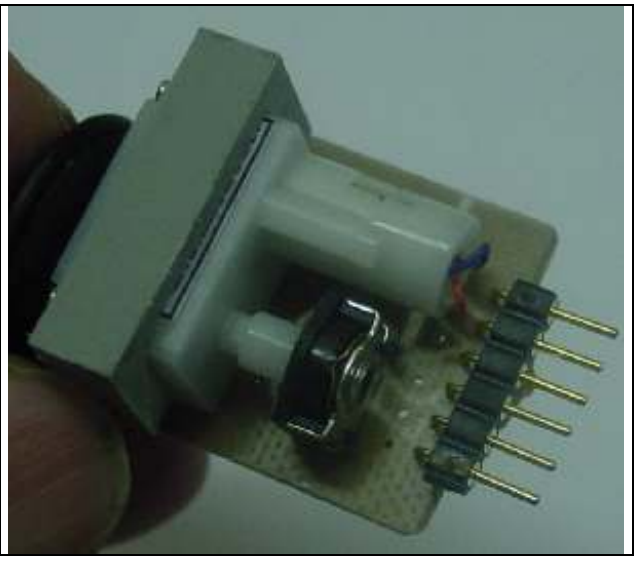

```
*/
#include "Ki2550Def.h"
#define IN0      PORTAbits.RA0
#define IN1      PORTAbits.RA1

enum {e0,e1,e2,e3} next = e0;
s16 pos=0 ;

void loop ()
{
        switch(next)
        {
            case e0:
                  if (IN0) {pos++; next = e1; break;}
                  if (IN1) {pos--; next = e3; break;}
                  break;
            case e1:
                  if (IN1) {pos++; next = e2; break;}
                  if (!IN0) {pos--; next = e0; break;}
                  break;
            case e2:
                  if (!IN0) {pos++; next = e3; break;}
                  if (!IN1) {pos--; next = e1; break;}
                  break;
            case e3:
                  if (!IN1) {pos++; next = e0; break;}
                  if (IN0) {pos--; next = e2; break;}
                  break;
            }
            PORTB = pos ;
            delay (5) ;
}
```

Comme exemple d'encodeur, le "Keith mini mechanical encoder" est 10mm de large et donne 48 transitions par tour. Lié à une roue de 30mm la resolution est de 2mm, acceptable pour un petit robot.



Didel propose le MiniRomEnco associé à un moteur Bo30.Un adapteur spécial a du être developpé pour relier le moteur et l'encodeur.

Le MiniRomEnco est compatible avec le RomEnco (a part que les rebonds n'existent pas sur le RomEnco qui a des capteurs de Hall).



Programme de test. La procédure s'appelle toutes les 20ms ou par interruption.

```
// \prog:TestMsEnco.ino|
Algorithme de Sommer
byte olda, oldb, newab ; // 0b000000AB
int position = 0 ;

void setup()  {
  DDRB = 0 ; // in
  DDRC = 0xFF ;
  olda = PINA ;
  oldb = PINB ;
}
```

```
//prog:TestMsEnco.ino suite

byte olda, oldb, newab ; // 0b000000AB
void setup()  {
  DDRB = 0 ; // in
  DDRC = 0xFF ;
  olda = PINB ;
  oldb = PINB ;
}

void loop() {
  newab = PINB & 0x03 ;
  olda = (olda>>1 & 0x01) ;
  oldb = (oldb<<1 & 0x02) ;
  olda |= oldb ;
  olda ^= newab ;
  if (olda == 0x01)   position++;
  if (olda == 0x02)   position--;
  olda = newab ;
  oldb = newab ;
    delay (10);
}
```

jdn 100500/131202