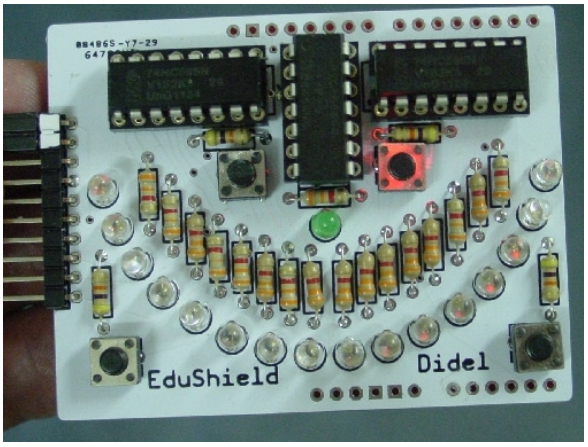
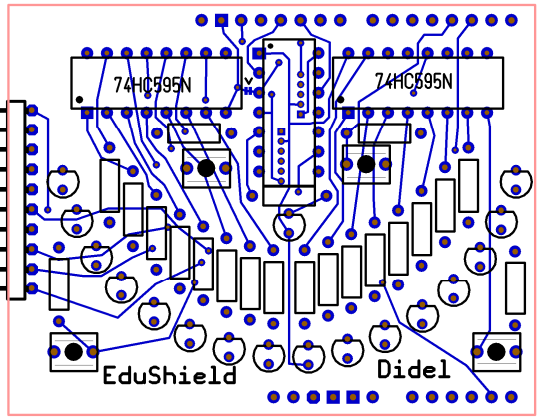
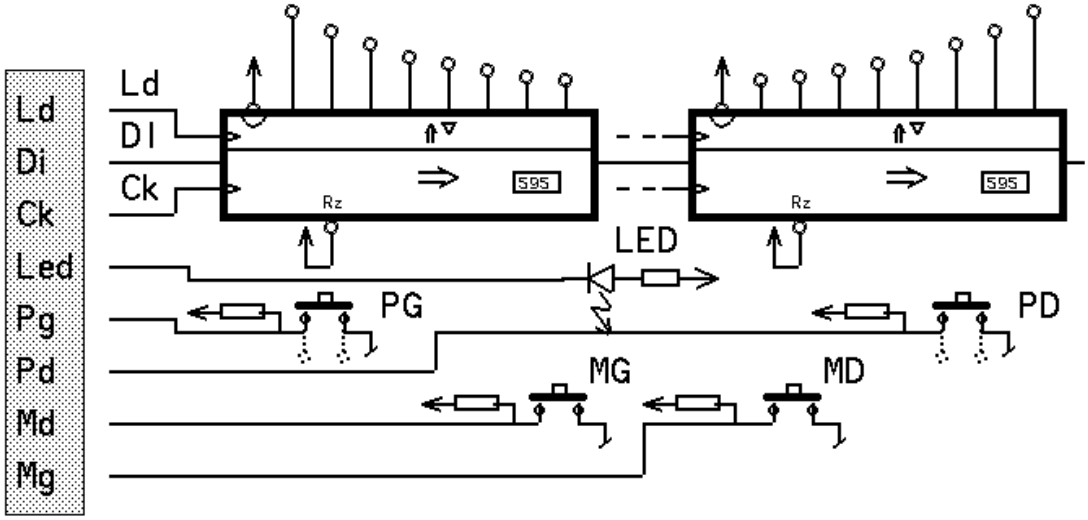
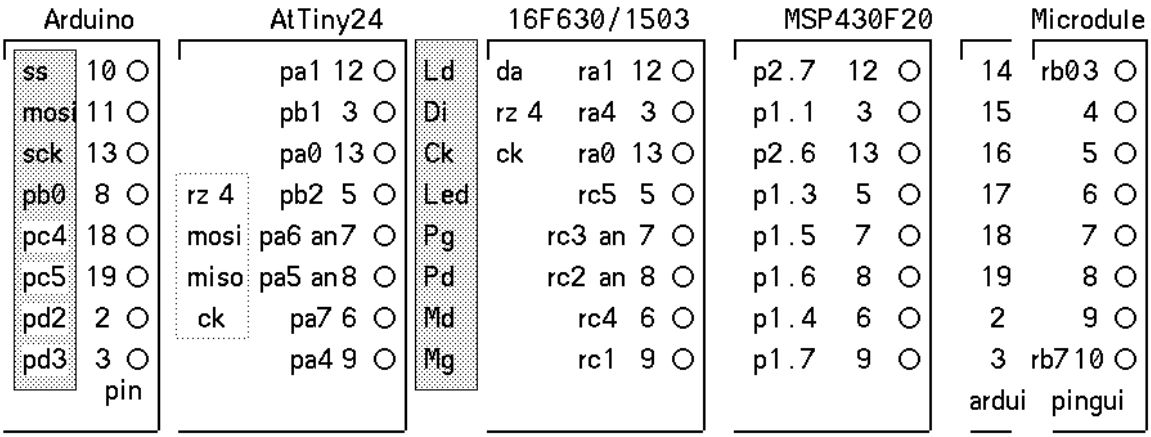
Newton

Une doc plus détaillée sera faite si intérêt. Un stagiaire a fait quelques programmes.









```
// LoicPong.ino
//ping-pong avec réglage de rapidité de la balle
//Didiuino + kidule Newton

// Define pour le sens
#define Gauche 0
#define Droite 1

      // Définition des pattes

int LATCH = 14;      // Pin de Latch pour le 595
int DATA = 15;       // Pin de Data pour le 595
int CLK = 16;        // Pin de Clock pour le 595
int LedVerte = 17;   // Pin pour la Led centrale
int CoinGauche = 18; // Pin pour le bouton en bas à gauche
int CoinDroite = 19; // Pin pour le bouton en bas à droite
int MidGauche = 2;   // Pin pour le bouton au millieu à gauche
int MidDroite = 3;   // Pin pour le bouton au millieu à droite

/*****************************************************************************
* Variables
*****************************************************************************/


unsigned long Drive16Led = 1; // Variable qui controle une led sur chaque bit
unsigned char Sens = Gauche;  // Donne le sens, début led droite sens gauche
//char Boutons = 0;           // Initialement pour contrôler les boutons
//char Presse = 0;    // Initialement pour dire quand un bouton est pressé(flan)
char i = 0;                   // Variable de comptage
char Bouge = 0;               // Pour faire changer l'état des leds quand on veut
char Tatt = 0;                // Temps d'attente des leds
char TTemps = 0;              // Variable de comptage dans ajustage
char AdjustTime = 0;          // Variable d'ajustement du temps
/*****************************************************************************
* Function    : Setup
*****************************************************************************


* Description : Routines de dÃ©finitions des entrÃ©es/sorties
*               Initialisations de diverses routines
*****************************************************************************/
void setup()
{
  pinMode(LATCH, OUTPUT);
  pinMode(CLK, OUTPUT);
  pinMode(DATA, OUTPUT);
  pinMode(LedVerte, OUTPUT);
  pinMode(CoinGauche, INPUT);
  pinMode(CoinDroite, INPUT);
  pinMode(MidGauche, INPUT);
  pinMode(MidDroite, INPUT);
  digitalWrite(LedVerte,HIGH); // On etteint la led centrale pour commencer
}

/*****************************************************************************
* Function    : loop
*****************************************************************************
* Description : Boucle du programme principal
*****************************************************************************/
void loop()
{                           // Sens gauche
        if((Sens == Gauche) && (Bouge == 1))
        {                    // Quand le sens est à gauche et qu'on peut bouger,
          Drive16Led = Drive16Led * 2;     // On passe à la led suivante
          SendToShiftRegister(~Drive16Led);// et on l'envoit au 595
          Bouge = 0;                       // sans oublier de remettre bouge à 0
        }

                            // Sens droite
        if((Sens == Droite)&&(Bouge == 1))
        {                     // Quand le sens est à droite et qu'on peut bouger,
```

```
          Drive16Led = Drive16Led / 2;     //On passe à la led précedente
          SendToShiftRegister(~Drive16Led);// et on l'envoit au 595
          Bouge = 0;                       // sans oublier de remettre bouge à 0
        }

                           // Raquette de droite
        if((digitalRead(CoinDroite) == 0) && (Sens == Droite))
        {  // Quand le bouton en bas à droite est pressé et le sens est à droite,
          if(Drive16Led == 0x001)
          {                  // et qu'on est sur la première led,
            Sens = Gauche;   // on envoit la "balle* dans l'autre sens
          }
          else
          {                  // Sinon on a perdu
            lose();

          }
        }
                          // Raquette de gauche
        if((digitalRead(CoinGauche) == 0) && (Sens == Gauche))
        { // Quand le bouton en bas à gauche est pressé et le sens est à gauche,
          if(Drive16Led == 0x8000)
          {                  // et qu'on est sur la dernière led,
            Sens = Droite;   // On envoit la "balle" dans l'autre sens
          }
          else
          {                  // Sinon on a perdu
            lose();
          }
        }

                          // Réglage temps diminution
        while(digitalRead(MidGauche) == 0)
        {                  // Lorsque l'on presse sur le bouton central gauche,
          TTemps ++;       // On incrémente la variable de comptage,
          delay(100);      // un passage de boucle = 100ms
          if(TTemps > 9)
          {                  // Quand on a attendu 900ms on augmente adjust,
            AdjustTime = AdjustTime + 1;    // ce qui baisse le temps de chaque
led
            digitalWrite(LedVerte,LOW);     // On allume la led verte pour 100ms,
            delay(100);                     // Pour indiquer quand le temps
change
            digitalWrite(LedVerte,HIGH);    // et on l'étteint
            TTemps = 0;    // Sans oublier de remettre à 0 la variable de
comptage
          }
        }
                          // Réglage temps augmentation
        while(digitalRead(MidDroite) == 0)
        {                  // Lorsque l'on presse sur le bouton central droite,
          TTemps ++;       // On incrémente la variable de comptage,
          delay(100);      // un passage de boucle = 100ms
          if(TTemps > 9)
          {                  // Quand on a attendu 900ms on baisse adjust,
            AdjustTime = AdjustTime - 1;// ce qui augmente le temps de chaque led
            digitalWrite(LedVerte,LOW);     // On allume la led verte pour 100ms,
            delay(100);    // Pour indiquer quand le temps change
            digitalWrite(LedVerte,HIGH);// et on l'étteint
            TTemps = 0;    // Sans oublier de remettre à 0 la variable de
comptage
          }
        }

                          // Balle sortie du terrain
        if(Drive16Led == 0)
        { // Quand la première ou la dernière led s'est étteinte, donc qu'on a
pas
          lose();               // pressé le bouton à temps,
        }                       // on a perdu
```

```
        DureeTemps(Drive16Led); // Réglage du temps de chaque led
}

void SendToShiftRegister(unsigned int Val)
{                               // Pour envoyer une donnée de 16 bits au 595,
  digitalWrite(LATCH, LOW);     //                 Il faut préparer le LATCH,
  shiftOut(DATA, CLK, LSBFIRST, (Val&0xFF));    // Envoyer le LSB
  shiftOut(DATA, CLK, LSBFIRST, (Val>>8)&0xFF); // Puis le MSB
  digitalWrite(LATCH, HIGH);// et ensuite faire un flan montant sur le latch
}

void DureeTemps(unsigned int Tmp)
{                   // Lors du réglage du temps de chaque led,
  char TLed;
  Tatt ++;          // Comptage du temps attendu
  for(i=0;i<16;i++)
  {                 // On regarde quel led est allumée,
    if((Tmp&1) == 1)
    {               // lorsque la bonne led est trouvée
      if(i<8)
      {             // et qu'elle se trouve dans le premier byte,
        TLed = (35-AdjustTime)-(i*4);// On utilise cette formule
      }
      if(i>7)
      {             // si elle se trouve dans le deuxième byte,
        TLed = ((35-AdjustTime)-((8-(i-7))*4));// On utilise celle-ci
      }
    }
    Tmp = Tmp >> 1; // décalage pour lire chaque bit séparément
  }
  if(Tatt > TLed)
  {                 // Lorsque le temps à attendre est atteint,
    Bouge = 1;      // On indique qu'on peut bouger les leds
    Tatt = 0;
  }
  delay(10);
}
void lose(void)
{                                 // Lorsque l'on a perdu,
  SendToShiftRegister(~0xFFFF);   // On allume toute les leds,
  digitalWrite(LedVerte,LOW);     // On allume la led verte
  delay(3000);                    // On attend 3 secondes
  Drive16Led = 1;                 // on recommencera la partie sur la doite
  digitalWrite(LedVerte,HIGH);    // on etteint la led verte
  Sens = Droite;
}
/*void LectureBoutons(void)
{
  Boutons |= (~digitalRead(CoinGauche))>>4;
  Boutons |= (~digitalRead(CoinDroite))>>5;
  Boutons |= (~digitalRead(MidGauche)) >>6;
  Boutons |= (~digitalRead(MidDroite)) >>7;
  Boutons |= ((Boutons & 0xF0)>>4)^(Boutons & 0x0F);
  Presse = (((Boutons & 0xF0)>>4)&(Boutons &0x0F));
  Boutons = (Boutons & 0xF0) >> 4;
}*/
```