Programmes Arduino pour 74HC595

Librairie Shift-register_74HC595

ShiftRegister74HC595 Bibliothèque Arduino

La bibliothèque Arduino ShiftRegister74HC595 simplifie l'utilisation des registres à décalage. Il vous permet de définir des broches individuelles de votre registre à décalage sur haut ou bas, tout comme les broches Arduino normales. Il supprime ainsi la surcharge de décalage des octets qui ont été créés avec des opérations compliquées au niveau des bits. Le réglage par exemple de la deuxième broche de votre registre à décalage ressemblerait simplement à

```
sr.set(2, HIGH);// mettre à 1 la sortie  2 du 74HC595
```

Prog1.ino

```
/*
  ShiftRegister74HC595 - Bibliothèque pour un contrôle simplifié des registres à décalage 74HC595.
   Développé et maintenu par Timo Denk et ses contributeurs, depuis novembre 2014.
   Des informations supplémentaires sont disponibles sur https://timodenk.com/blog/shift-register-arduino-library/
   Lâché dans le domaine public.
*/

#include <ShiftRegister74HC595.h>

// create a global shift register object
// parameters: <number of shift registers> (data pin, clock pin, latch pin)
ShiftRegister74HC595<1> sr(0, 1, 2);

void setup() {
}

void loop() {

  // setting all pins at the same time to either HIGH or LOW
  sr.setAllHigh(); // set all pins HIGH
  delay(500);

  sr.setAllLow(); // set all pins LOW
  delay(500);


  // setting single pins
```

```
  for (int i = 0; i < 8; i++) {

    sr.set(i, HIGH); // set single pin HIGH
    delay(250);
  }


  // set all pins at once
  uint8_t pinValues[] = { B10101010 };
  sr.setAll(pinValues);
  delay(1000);


  // read pin (zero based, i.e. 6th pin)
  uint8_t stateOfPin5 = sr.get(5);
  sr.set(6, stateOfPin5);


  // set pins without immediate update
  sr.setNoUpdate(0, HIGH);
  sr.setNoUpdate(1, LOW);
  // at this point of time, pin 0 and 1 did not change yet
  sr.updateRegisters(); // update the pins to the set values
}
```

Programmation d un 74HC595 avec la librairie Shifty

Librairie Shifty

Un gestionnaire 74HC595 flexible pour Arduino

La bibliothèque Shifty pour Arduino est un moyen très flexible de gérer les registres à décalage 74HC595. Il vous permet d'écrire sur des sorties individuelles, tout comme “digitalWrite”, vous permet de connecter en guirlande des registres à décalage et, si vous le câblez conformément aux instructions de ce document, vous permet d'utiliser votre registre à décalage pour l' entrée et la sortie. broches avec une seule broche supplémentaire utilisée. Cela le rend idéal pour une utilisation avec un ATTiny, bien qu'il utilise un peu d'espace sur l'appareil.

Prog_Test_Shifty

```
#include <Shifty.h>

// Declare the shift register
Shifty shift;

void setup() {
  // Set the number of bits you have (multiples of 8)
```

```
  shift.setBitCount(8);

  // Set the clock, data, and latch pins you are using
  // This also sets the pinMode for these pins
  shift.setPins(11, 12, 8);
}

void loop() {
  // writeBit works just like digitalWrite
  shift.writeBit(1, HIGH);
  delay(500);
  shift.writeBit(3, HIGH);
  delay(500);
  shift.writeBit(1, LOW);
  delay(500);
  shift.writeBit(3, LOW);

  delay(500);
}
```

Programme pour 74HC595 sans librairie

Prog2.ino

```
int SER_Pin = 8;   //pin 14 sur le 75HC595
int RCLK_Pin = 9;  //pin 12 sur le 75HC595
int SRCLK_Pin = 10; //pin 11 sur le 75HC595

#define number_of_74hc595s 1 //Combien combinez-vous de 74HC595 ? Ne
pas toucher si 1 seul

#define numOfRegisterPins number_of_74hc595s * 8
boolean registers[numOfRegisterPins];
void setup(){
pinMode(SER_Pin, OUTPUT);
pinMode(RCLK_Pin, OUTPUT);
pinMode(SRCLK_Pin, OUTPUT);
//Reset tous les pins du 74HC595
clearRegisters();
writeRegisters();
}
//Place tous les pins du 74HC595 à l'état "OFF"
void clearRegisters(){
for(int i = numOfRegisterPins - 1; i >=  0; i--){
registers[i] = LOW;
}
}
//Enregistrer et afficher les valeurs dans le registre
```

```
//Executer uniquement APRES que toutes les valeurs aient été
programmées
void writeRegisters(){
digitalWrite(RCLK_Pin, LOW);
for(int i = numOfRegisterPins - 1; i >=  0; i--){
digitalWrite(SRCLK_Pin, LOW);
int val = registers[i];
digitalWrite(SER_Pin, val);
digitalWrite(SRCLK_Pin, HIGH);
}
digitalWrite(RCLK_Pin, HIGH);
}
//Place un pin du 74HC595 à l'état HAUT ou BAS
void setRegisterPin(int index, int value){
registers[index] = value;
}
void loop(){
setRegisterPin(2, HIGH);
setRegisterPin(3, HIGH);
setRegisterPin(4, LOW);
setRegisterPin(5, HIGH);
setRegisterPin(7, HIGH);
writeRegisters();  //Doit être exécuté pour appliquer les changements
//Executer seulement une fois que toutes les valeurs ont été
enregistrées comme vous le souhaitiez.
}
```

Programmation 74HC595 en utilisant les bits du registre plus rapide mais plus compliqué

Prog3.ino

```
// Broche connectée au ST_CP du 74HC595
const int verrou = 11;
// Broche connectée au SH_CP du 74HC595
const int horloge = 12;
// Broche connectée au DS du 74HC595
const int data = 10;

void setup()
{
    // On met les broches en sortie
    pinMode(verrou, OUTPUT);
    pinMode(horloge, OUTPUT);
    pinMode(data, OUTPUT);
}
```

```
void loop()
{
    // on affiche les nombres de 0 à 255 en binaire
    for (char i = 0; i<256; i++)
    {
        // On active le verrou le temps de transférer les données
        digitalWrite(verrou, LOW);
        // on envoi toutes les données grâce à notre belle fonction
        envoi_ordre(data, horloge, 1, ~i);
        // et enfin on relâche le verrou
        digitalWrite(verrou, HIGH);
        // une petite pause pour constater l'affichage
        delay(1000);
    }
}

void envoi_ordre(int dataPin, int clockPin, boolean sens, char donnee)
{
    // on va parcourir chaque bit de l'octet
    for(int i=0; i<8; i++)
    {
        // on met l'horloge à l'état bas
        digitalWrite(clockPin, LOW);
        // on met le bit de donnée courante en place
        if(sens)
        {
            digitalWrite(dataPin, donnee & 0x01 << i);
        }
        else
        {
            digitalWrite(dataPin, donnee & 0x80 >> i);
        }
        // enfin on remet l'horloge à l'état haut pour
        // faire prendre en compte cette dernière
        digitalWrite(clockPin, HIGH);
    }
}
```