```
/**************************************
    スタートゲート制御基板用スケッチ
    LCD Keypad Shield 使用
 **************************************/

//Sample using LiquidCrystal library
#include <LiquidCrystal.h>
#include <Boards.h>
#include <Firmata.h>
#include <Servo.h>

/*select the pins used on the LCD panel
      lcd の使っているピン番号
   LiquidCrystal(rs, enable, d4, d5, d6, d7)
   rs: LCD の RS ピンに接続する Arduino 側のピン番号
   rw: LCD の RW ピンに接続する Arduino 側のピン番号
   enable: LCD の enable ピンに接続する Arduino 側のピン番号
   d0～d7: LCD の data ピンに接続する Arduino 側のピン番号

   d0～d3 はオプションで、省略すると 4 本のデータライン(d4～d7)だけで制御します。 */
LiquidCrystal lcd(8, 9, 4, 5, 6, 7);

// define some values used by the panel and buttons
int pushbutton = 0;

//サーボ
Servo myservo1;
Servo myservo2;
const int SERVO1OPEN = 127; //OPEN と CLOSE が逆
const int SERVO1CLOSE = 40;
const int SERVO2OPEN = 120;
const int SERVO2CLOSE = 35;

//パターン
int INTimepattern = 4;
int OUTTimepattern = 4;
int pattern = 0;
long INMainpattern = 0;
long OUTMainpattern = 0;
int INServopattern = 0;
```

```
int OUTServopattern = 0;

/*I = IN
   O = OUT
   cnts = second
   cntm = minute
   分秒表示*/
int Icnts = 0;
int Icntm = 0;
int Ocnts = 0;
int Ocntm = 0;

//時間計算
unsigned long Itimemillis = 0;
unsigned long Otimemillis = 0;
unsigned long Itime1 = 0;
unsigned long Itime2 = 0;
unsigned long Itime3 = 0;
unsigned long Otime1 = 0;
unsigned long Otime2 = 0;
unsigned long Otime3 = 0;

//時間
unsigned long millis();
unsigned long icnt = 0;
unsigned long ocnt = 0;

//時間その他
int IcntIN = 0;
int OcntOUT = 0;
unsigned long OT = 0;
unsigned long IT = 0;

//proccesing 変数宣言
int outTime_m = 0; // OUT コース:分
int outTime_s = 0; // OUT コース:秒
int outTime_c = 0; // OUT コース:ミリ秒
int inTime_m = 0;   // IN コース:分
int inTime_s = 0;   // IN コース:秒
int inTime_c = 0;   // IN コース:ミリ秒
```

```
//タイマー表示 2 周目 3 周目
int secondOTIME = 0;
int thirdOTIME = 0;
int secondITIME = 0;
int thirdITIME = 0;
int secondOcntM = 0;
int secondOcntS = 0;
int thirdOcntM = 0;
int thirdOcntS = 0;
int secondIcntM = 0;
int secondIcntS = 0;
int thirdIcntM = 0;
int thirdIcntS = 0;

//定義
#define LEFT     0
#define UP       1
#define DOWN     2
#define RIGHT    3
#define SELECT 4
#define NONE     5

//LCD キーパッドシールドスイッチ入力
#define analogswitch A0

//センサー
#define sensor1 2     //1 41
#define sensor2 17    //1 41D
#define sensor3 3     //2 41
#define sensor4 18    //2 41D
int sensorin1;
int sensorin2;
int sensorin3;
int sensorin4;
//A0～A5→D14～D19 変換可能
volatile int sens1;
volatile int sens2;

// read the buttons
```

```
int button() {
  pushbutton = (analogRead(analogswitch) / 4);
  if (pushbutton >= 240) return NONE;//240
  if (pushbutton < 20) return   RIGHT;//20
  if (pushbutton < 70) return   UP;//70
  if (pushbutton < 120) return DOWN;//120
  if (pushbutton < 170) return LEFT;//170
  if (pushbutton < 240) return SELECT;//240
  //return NONE;
}


void timerIN() {
  pattern = button();
  switch (INTimepattern) {
    case 0:
      Itimemillis = millis();
      Itime2 = Itimemillis - Itime1;
      if (Itime2 > 999) {
        Itime1 = Itimemillis;
        Icnts += 1;
      }
      if (Icnts > 59) {
        Icnts = 0;
        Icntm += 1;
      }
      //if (pattern == UP && IT > 500)
      if (INMainpattern == 21) {
        secondITIME = Itime2;
        secondIcntS = Icnts;
        secondIcntM = Icntm;
        inTime_c = secondITIME / 10;
        inTime_s = secondIcntS;
        inTime_m = secondIcntM;
      }
      IT++;
      IcntIN = 0;
      break;

    case 2:
```

```
        IcntIN++;
        break;

      case 4:
        IcntIN++;
        Itimemillis = millis();
        Itime1 = Itimemillis;
        Itime2 = Itimemillis - Itime1;
        Icnts = 0;
        Icntm = 0;
        secondITIME = 0;
        secondIcntS = 0;
        secondIcntM = 0;
        inTime_c = secondITIME;
        inTime_s = secondIcntS;
        inTime_m = secondIcntM;
        break;
    }
}

void timerOUT() {
    pattern = button();
    switch (OUTTimepattern) {
      case 0:
        Otimemillis = millis();
        Otime2 = Otimemillis - Otime1;
        if (Otime2 > 999) {
          Otime1 = Otimemillis;
          Ocnts += 1;
        }
        if (Ocnts > 59) {
          Ocnts = 0;
          Ocntm += 1;
        }
        //if (pattern == RIGHT && OT > 500)
        if (OUTMainpattern == 21) {
          secondOTIME = Otime2;
          secondOcntS = Ocnts;
          secondOcntM = Ocntm;
          outTime_c = secondOTIME / 10;
```

```
        outTime_s = secondOcntS;
        outTime_m = secondOcntM;
      }
      OT++;
      OcntOUT = 0;
      break;

    case 2:
      OcntOUT++;
      break;

    case 4:
      OcntOUT++;
      Otimemillis = millis();
      Otime1 = Otimemillis;
      Otime2 = Otimemillis - Otime1;
      Ocnts = 0;
      Ocntm = 0;
      secondOTIME = 0;
      secondOcntS = 0;
      secondOcntM = 0;
      outTime_c = secondOTIME;
      outTime_s = secondOcntS;
      outTime_m = secondOcntM;
      break;
  }
}

void Sensor1() {//1
  if (digitalRead(sensor1) == HIGH) {
    sens1 = 1;
  }
  if (digitalRead(sensor1) == LOW) {
    sens1 = 0;
  }
}

void Sensor2() {//2
  if (digitalRead(sensor3) == HIGH) {
    sens2 = 1;
```

```
  }
  if (digitalRead(sensor3) == LOW) {
    sens2 = 0;
  }
}

void MainIN() {
  Sensor2();
  pattern = button();   // read the buttons
  switch (INMainpattern) {
    case 0:
      INServopattern = 1;
      if (pattern == UP) {
        icnt = 0;
        INServopattern = 0;
        INMainpattern = 10;//大会用
        break;
      }
      if (pattern == DOWN) {
        icnt = 0;
        INMainpattern = 500;//フリー走行用
        break;
      }
      break;

    case 10://スタンバイ
      icnt++;
      if (pattern == LEFT && icnt >= 100) {
        INTimepattern = 0;
        icnt = 0;
        INServopattern = 1;
        INMainpattern = 20;
        break;
      }

    case 20://分岐
      icnt++;
      if (sens2 == 1 && icnt >= 200) {
        icnt = 0;
        INMainpattern = 21;//通過処理
```

```
            break;
        }
        if (pattern == RIGHT && icnt >= 300) {
            icnt = 0;
            INServopattern = 0;
            INTimepattern = 2;
            INMainpattern = 50;
            break;
        }
        break;

    case 21://通過
        icnt++;
        if (icnt >= 1) {
            icnt = 0;
            INMainpattern = 20;
            break;
        }
        break;

    case 50://終了
        icnt++;
        if (pattern == RIGHT && icnt >= 300) {
            icnt = 0;
            INTimepattern = 4;
            INMainpattern = 10;
            break;
        }
        break;
  }
}

void MainOUT() {
  Sensor1();
  pattern = button();   // read the buttons
  switch (OUTMainpattern) {
    case 0:
      OUTServopattern = 1;
      if (pattern == UP) {
        ocnt = 0;
```

```
      OUTServopattern = 0;
      OUTMainpattern = 10;//大会用
      break;
    }
    if (pattern == DOWN) {
      ocnt = 0;
      OUTMainpattern = 500;//フリー走行用
      break;
    }
    break;

  case 10://スタンバイ
    ocnt++;
    if (pattern == LEFT && ocnt >= 100) {
      OUTTimepattern = 0;
      ocnt = 0;
      OUTServopattern = 1;
      OUTMainpattern = 20;
      break;
    }
    break;

  case 20://分岐
    ocnt++;
    if (sens1 == 1 && ocnt >= 200) {
      ocnt = 0;
      OUTMainpattern = 21;//通過処理
      break;
    }
    if (pattern == RIGHT && ocnt >= 300) {
      ocnt = 0;
      OUTServopattern = 0;
      OUTTimepattern = 2;
      OUTMainpattern = 50;
      break;
    }
    break;

  case 21://通過
    ocnt++;
```

```
        if (ocnt >= 1) {
          ocnt = 0;
          OUTMainpattern = 20;
          break;
        }
        break;

      case 50://終了
        ocnt++;
        if (pattern == RIGHT && ocnt >= 300) {
          ocnt = 0;
          OUTTimepattern = 4;
          OUTMainpattern = 10;
          break;
        }
        break;
    }
}

void Servo1() {
  switch (OUTServopattern) {
    case 0:
      myservo1.write(SERVO1OPEN);
      break;

    case 1:
      myservo1.write(SERVO1CLOSE);
      break;
  }
}

void Servo2() {
  switch (INServopattern) {
    case 0:
      myservo2.write(SERVO2OPEN);
      break;

    case 1:
      myservo2.write(SERVO2CLOSE);
      break;
```

```
    }
}

void LCD() {
    lcd.setCursor(5, 1);
    lcd.print(Icntm);
    lcd.print(" ");
    lcd.print(Icnts);
    lcd.print(" ");
    lcd.print(Itime2);
    lcd.print("    ");
    lcd.setCursor(5, 0);
    lcd.print(Ocntm);
    lcd.print(" ");
    lcd.print(Ocnts);
    lcd.print(" ");
    lcd.print(Otime2);
    lcd.print("    ");
}

void LCD2() {
    lcd.setCursor(5, 1);
    lcd.print(secondIcntM);
    lcd.print(" ");
    lcd.print(secondIcntS);
    lcd.print(" ");
    lcd.print(secondITIME);
    lcd.print("    ");
    lcd.setCursor(5, 0);
    lcd.print(secondOcntM);
    lcd.print(" ");
    lcd.print(secondOcntS);
    lcd.print(" ");
    lcd.print(secondOTIME);
    lcd.print("    ");
}

void proccesing() {
    timerIN();
    timerOUT();
```

```
  Serial.print("H");         // ヘッダ送信(先頭を示す文字)
  Serial.write(outTime_c);// OUT コース;センチ秒データ送信
  Serial.write(inTime_c);// IN コースミリ秒データ送信
  Serial.write(outTime_m);// OUT コース;分データ送信
  Serial.write(outTime_s);// OUT コース;秒データ送信
  Serial.write(inTime_m); // IN コース;分データ送信
  Serial.write(inTime_s); // IN コース;秒データ送信
  Serial.print('¥n');
}

void setup()
{
  Serial.begin(250000);
  pinMode(sensor1, INPUT_PULLUP);
  pinMode(sensor2, INPUT_PULLUP);
  pinMode(sensor3, INPUT_PULLUP);
  pinMode(sensor4, INPUT_PULLUP);
  attachInterrupt(0, Sensor1, CHANGE);
  attachInterrupt(1, Sensor2, CHANGE);
  myservo1.attach(12);
  myservo2.attach(13);
  lcd.begin(16, 2);                    // start the library
  lcd.setCursor(0, 0);
  lcd.print("OUT"); // print a simple message
  lcd.setCursor(0, 1);    // move to the begining of the second line
  lcd.print("IN");
}

void loop() {
  timerIN();
  timerOUT();
  MainIN();
  MainOUT();
  Servo1();
  Servo2();
  LCD2();
  proccesing();
```