EEPROM 版

```
/*
* Uno   : input -> A0, HWIR -> D2, SCL -> A5, SDA -> A4
* 32ubb :          F7,         E6,        D0,        D1
*/
#define KEYBOARD_MODE 1

#include <Wire.h>

const byte EEPROM_ID = 0x50; // I2C address for 24LC256 ( A0 = A1 = A2 = GND )

const int analogInPin0 = 0; // Analog In 0 @ Uno, F7 @ 32ubb
const int irRecievePin = 7; // Hardware Interrupt : D2 = 2 @ Uno, E6 = 7 @ 32ubb
const int irNo = 4;         //                    : 0      @ Uno, 4      @ 32ubb
int sensorValue = 0;

volatile int repeat;
unsigned int rep0, rep1, rep2, rep3;
String seq0, seq1, seq2, seq3;

/***  Setup  ***/
void setup()
{
 Serial.begin( 9600 );

 //while ( !Serial ) ; // wait for serial port to connect. Needed for Leonardo only
 Wire.begin();

 String buf0, buf1, buf2, buf3;
 byte c;
 int split_cnt = 0;

 //Serial.println( "start reading config from EEPROM" );
 for ( int i = 0; i < 1024; i++ ) {

   c = i2cEEPROM_Read( i );
   if ( char( c ) == '|' ) {
     split_cnt++;
     continue;
   }

   if      ( split_cnt < 1 ) buf0 += char( c );
   else if ( split_cnt < 2 ) seq0 += char( c );
   else if ( split_cnt < 3 ) buf1 += char( c );
   else if ( split_cnt < 4 ) seq1 += char( c );
   else if ( split_cnt < 5 ) buf2 += char( c );
   else if ( split_cnt < 6 ) seq2 += char( c );
   else if ( split_cnt < 7 ) buf3 += char( c );
   else if ( split_cnt < 8 ) seq3 += char( c );
 }

 rep0 = buf0.toInt();
 rep1 = buf1.toInt();
 rep2 = buf2.toInt();
 rep3 = buf3.toInt();
 /*
 Serial.println( rep0 );  Serial.println( seq0 );
 Serial.println( rep1 );  Serial.println( seq1 );
 Serial.println( rep3 );  Serial.println( seq3 );
 Serial.println( rep2 );  Serial.println( seq2 );

 Serial.println( "read config from EEPROM." );
 */
 pinMode( irRecievePin, INPUT );
 attachInterrupt( irNo, stop_cmd, CHANGE );
}

/***  Main Loop  ***/
void loop()
{
 String sequence;
 repeat = 0;
 sensorValue = analogRead( analogInPin0 );
 //Serial.println( sensorValue ); delay( 50 );

 // no button pushed, sensorValue : 1023
 if ( sensorValue < 50 ) {        // button 3 pushed : 862
   sequence = seq3;
```

```
  repeat   = rep3;
 }
 else if ( sensorValue < 660 ) {  // button 2 pushed : 434
   sequence = seq2;
   repeat   = rep2;
 }
 else if ( sensorValue < 780 ) {  // button 1 pushed : 215
   sequence = seq1;
   repeat   = rep1;
 }
 else if ( sensorValue < 880 ) {  // button 0 pushed : 212
   sequence = seq0;
   repeat   = rep0;
 }

 //
 String buf = sequence;
 while ( repeat > 0 ) {

     String cmdline;
     boolean more = true;
     int n;
     sequence = buf;

     while ( more ) {
       n = sequence.indexOf( '\n' );
       cmdline = sequence.substring( 0, n );
       sequence = sequence.substring( n + 1 );
       more = send_key( cmdline );
     }
     delay( 100 );
     repeat--;
 }

 delay( 50 );
}

/***  Subroutines  ***/
/* read from external memory */
byte i2cEEPROM_Read( unsigned int address )
{
   byte data;
   Wire.beginTransmission( EEPROM_ID );
   Wire.write( (int)highByte(address) );
   Wire.write( (int)lowByte(address) );
   Wire.endTransmission();
   Wire.requestFrom( EEPROM_ID, (byte)1 );
   while ( Wire.available() == 0 )
       ;
   data = Wire.read();
   return data;
}

/*  send keycode or move cursor  */
boolean send_key( String cmd )
{
 String key;
 char c;

 if ( cmd.equals( "__END__" ) )
   return false;

 if ( cmd.startsWith( "wait " ) ) {
   String t = cmd.substring( 5 );
   Serial.println( "wait " + t + " ..." );
   for ( int i = 0; i < 100; i++ )
     delay( t.toInt() );
 }

 // Mouse event
 else if ( cmd.startsWith( "move " ) ) {
   int n = cmd.indexOf( ',' );
   String x = cmd.substring( 5, n );
   String y = cmd.substring( n + 1 );
   if ( KEYBOARD_MODE )
     Mouse.move( x.toInt(), y.toInt() );
   else
     Serial.println( "move " + x + ", " + y );
 }
 else if ( cmd.equals( "click" ) ) {
   if ( KEYBOARD_MODE )
     Mouse.click();
```

```
    else
      Serial.println( "click" );
  }

  // Keyboard event
  else if ( cmd.equals( "releaseAll" ) ) {
    if ( KEYBOARD_MODE )
      Keyboard.releaseAll();
    else
      Serial.println( "releaseAll" );
  }
  else if ( cmd.startsWith( "press " ) ) {
    key = cmd.substring( 6 );
    if ( KEYBOARD_MODE ) {
      c = str2char( key );
      Keyboard.press( c );
    }
    else
      Serial.println( key );
  }
  else if ( cmd.startsWith( "write " ) ) {
    key = cmd.substring( 6 );
    if ( KEYBOARD_MODE ) {
      c = str2char( key );
      Keyboard.write( c );
    }
    else
      Serial.println( key );
  }
  else if ( cmd.startsWith( "print " ) ) {
    key = cmd.substring( 6 );
    if ( KEYBOARD_MODE )
      Keyboard.print( key );
    else
      Serial.println( key );
  }
  else if ( cmd.startsWith( "println " ) ) {
    key = cmd.substring( 8 );
    if ( KEYBOARD_MODE )
      Keyboard.println( key );
    else
      Serial.println( key );
  }

  return true;
}

char str2char( String s )
{
  if ( s.equals( "KEY_SPACE"      ) ) return 0x20;
  if ( s.equals( "KEY_LEFT_CTRL"  ) ) return KEY_LEFT_CTRL;
  if ( s.equals( "KEY_LEFT_SHIFT" ) ) return KEY_LEFT_SHIFT;
  if ( s.equals( "KEY_LEFT_ALT"   ) ) return KEY_LEFT_ALT;
  if ( s.equals( "KEY_TAB"        ) ) return KEY_TAB;
  if ( s.equals( "KEY_RETURN"     ) ) return KEY_RETURN;
  if ( s.equals( "KEY_F1"         ) ) return KEY_F1;
  if ( s.equals( "KEY_F2"         ) ) return KEY_F2;
  if ( s.equals( "KEY_F3"         ) ) return KEY_F3;
  if ( s.equals( "KEY_F4"         ) ) return KEY_F4;
  if ( s.equals( "KEY_F5"         ) ) return KEY_F5;
  if ( s.equals( "KEY_F6"         ) ) return KEY_F6;
  if ( s.equals( "KEY_F7"         ) ) return KEY_F7;
  if ( s.equals( "KEY_F8"         ) ) return KEY_F8;
  if ( s.equals( "KEY_F9"         ) ) return KEY_F9;
  if ( s.equals( "KEY_F10"        ) ) return KEY_F10;
  if ( s.equals( "KEY_F11"        ) ) return KEY_F11;
  if ( s.equals( "KEY_F12"        ) ) return KEY_F12;
  return KEY_ESC;
}

/*  hardware interrupt  */
void stop_cmd()
{
  repeat = -1;
}
```

シーケンス組み込み

```
/*
```

```
* Uno   : input -> A0, HWIR -> D2, SCL -> A5, SDA -> A4
* 32ubb :          F7,         E6,        D0,        D1
*/
#define KEYBOARD_MODE 1

const int analogInPin0 = 0; // Analog In 0 @ Uno, F7 @ 32ubb
const int irRecievePin = 7; // Hardware Interrupt : D2 = 2 @ Uno, E6 = 7 @ 32ubb
const int irNo = 4;         //                    : 0      @ Uno, 4      @ 32ubb
int sensorValue = 0;

volatile int repeat;
unsigned int rep0, rep1, rep2, rep3;
String seq0, seq1, seq2, seq3;

/***  Setup  ***/
void setup()
{
 String s;
 s += "2";
 s += "|";
 s += "press KEY_LEFT_SHIFT\n";
 s += "press KEY_F12\n";
 s += "releaseAll\n";
 s += "wait 450\n";
 s += "print 21200005\n";
 s += "print 0005\n";
 s += "write KEY_TAB\n";
 s += "write KEY_RETURN\n";
 s += "__END__\n";

 s += "|";
 s += "1";
 s += "|";
 s += "press KEY_LEFT_SHIFT\n";
 s += "press KEY_F12\n";
 s += "releaseAll\n";
 s += "wait 450\n";
 s += "print 21500005\n";
 s += "print 0005\n";
 s += "write KEY_TAB\n";
 s += "write KEY_RETURN\n";
 s += "__END__\n";

 s += "|";
 s += "1";
 s += "|";
 s += "press KEY_LEFT_SHIFT\n";
 s += "press KEY_F12\n";
 s += "releaseAll\n";
 s += "wait 450\n";
 s += "print 43400005\n";
 s += "print 0005\n";
 s += "write KEY_TAB\n";
 s += "write KEY_RETURN\n";
 s += "__END__\n";

 s += "|";
 s += "10";
 s += "|";
 s += "press KEY_LEFT_SHIFT\n";
 s += "press KEY_F12\n";
 s += "releaseAll\n";
 s += "wait 450\n";
 s += "print 86200005\n";
 s += "print 0005\n";
 s += "write KEY_TAB\n";
 s += "write KEY_RETURN\n";
 s += "__END__\n";
 s += "|";


 String buf0, buf1, buf2, buf3;
 int split_cnt = 0;

 for ( int i = 0; i < s.length(); i++ ) {

   if ( s[i] == '|' ) {
     split_cnt++;
     continue;
   }

   if      ( split_cnt < 1 ) buf0 += s[i];
```

```
    else if ( split_cnt < 2 ) seq0 += s[i];
    else if ( split_cnt < 3 ) buf1 += s[i];
    else if ( split_cnt < 4 ) seq1 += s[i];
    else if ( split_cnt < 5 ) buf2 += s[i];
    else if ( split_cnt < 6 ) seq2 += s[i];
    else if ( split_cnt < 7 ) buf3 += s[i];
    else if ( split_cnt < 8 ) seq3 += s[i];
  }

  rep0 = buf0.toInt();
  rep1 = buf1.toInt();
  rep2 = buf2.toInt();
  rep3 = buf3.toInt();

  Serial.begin( 9600 );

  while ( !Serial ) ; // wait for serial port to connect. Needed for Leonardo only
  Serial.println( rep0 );  Serial.println( seq0 );
  Serial.println( rep1 );  Serial.println( seq1 );
  Serial.println( rep2 );  Serial.println( seq2 );
  Serial.println( rep3 );  Serial.println( seq3 );

  pinMode( irRecievePin, INPUT );
  attachInterrupt( irNo, stop_cmd, CHANGE );
}

/***  Main Loop  ***/
void loop()
{
  String sequence;
  repeat = 0;
  sensorValue = analogRead( analogInPin0 );

  Serial.println( sensorValue ); delay( 50 );

  // no button pushed, sensorValue : 1023
  if ( sensorValue < 50 ) {        // button 3 pushed : 862
    sequence = seq3;
    repeat   = rep3;
  }
  else if ( sensorValue < 660 ) {  // button 2 pushed : 434
    sequence = seq2;
    repeat   = rep2;
  }
  else if ( sensorValue < 780 ) {  // button 1 pushed : 215
    sequence = seq1;
    repeat   = rep1;
  }
  else if ( sensorValue < 880 ) {  // button 0 pushed : 212
    sequence = seq0;
    repeat   = rep0;
  }

  //
  String buf = sequence;
  while ( repeat > 0 ) {

      String cmdline;
      boolean more = true;
      int n;
      sequence = buf;

      while ( more ) {
        n = sequence.indexOf( '\n' );
        cmdline = sequence.substring( 0, n );
        sequence = sequence.substring( n + 1 );
        more = send_key( cmdline );
      }
      delay( 100 );
      repeat--;
  }

  delay( 50 );
}

/***  Subroutines  ***/
/*  send keycode or move cursor  */
boolean send_key( String cmd )
{
  String key;
  char c;
```

```
 if ( cmd.equals( "__END__" ) )
   return false;

 if ( cmd.startsWith( "wait " ) ) {
   String t = cmd.substring( 5 );
   Serial.println( "wait " + t + " ..." );
   delay( t.toInt() );
 }

 // Mouse event
 else if ( cmd.startsWith( "move " ) ) {
   int n = cmd.indexOf( ',' );
   String x = cmd.substring( 5, n );
   String y = cmd.substring( n + 1 );
   if ( KEYBOARD_MODE )
     Mouse.move( x.toInt(), y.toInt() );
   else
     Serial.println( "move " + x + ", " + y );
 }
 else if ( cmd.equals( "click" ) ) {
   if ( KEYBOARD_MODE )
     Mouse.click();
   else
     Serial.println( "click" );
 }

 // Keyboard event
 else if ( cmd.equals( "releaseAll" ) ) {
   if ( KEYBOARD_MODE )
     Keyboard.releaseAll();
   else
     Serial.println( "releaseAll" );
 }
 else if ( cmd.startsWith( "press " ) ) {
   key = cmd.substring( 6 );
   if ( KEYBOARD_MODE ) {
     c = str2char( key );
     Keyboard.press( c );
   }
   else
     Serial.println( key );
 }
 else if ( cmd.startsWith( "write " ) ) {
   key = cmd.substring( 6 );
   if ( KEYBOARD_MODE ) {
     c = str2char( key );
     Keyboard.write( c );
   }
   else
     Serial.println( key );
 }
 else if ( cmd.startsWith( "print " ) ) {
   key = cmd.substring( 6 );
   if ( KEYBOARD_MODE )
     Keyboard.print( key );
   else
     Serial.println( key );
 }
 else if ( cmd.startsWith( "println " ) ) {
   key = cmd.substring( 8 );
   if ( KEYBOARD_MODE )
     Keyboard.println( key );
   else
     Serial.println( key );
 }

 return true;
}

char str2char( String s )
{
 if ( s.equals( "KEY_LEFT_CTRL"  ) ) return KEY_LEFT_CTRL;
 if ( s.equals( "KEY_LEFT_SHIFT" ) ) return KEY_LEFT_SHIFT;
 if ( s.equals( "KEY_LEFT_ALT"   ) ) return KEY_LEFT_ALT;
 if ( s.equals( "KEY_TAB"        ) ) return KEY_TAB;
 if ( s.equals( "KEY_RETURN"     ) ) return KEY_RETURN;
 if ( s.equals( "KEY_F1"         ) ) return KEY_F1;
 if ( s.equals( "KEY_F2"         ) ) return KEY_F2;
 if ( s.equals( "KEY_F3"         ) ) return KEY_F3;
 if ( s.equals( "KEY_F4"         ) ) return KEY_F4;
 if ( s.equals( "KEY_F5"         ) ) return KEY_F5;
 if ( s.equals( "KEY_F6"         ) ) return KEY_F6;
```

```
 if ( s.equals( "KEY_F7"         ) ) return KEY_F7;
 if ( s.equals( "KEY_F8"         ) ) return KEY_F8;
 if ( s.equals( "KEY_F9"         ) ) return KEY_F9;
 if ( s.equals( "KEY_F10"        ) ) return KEY_F10;
 if ( s.equals( "KEY_F11"        ) ) return KEY_F11;
 if ( s.equals( "KEY_F12"        ) ) return KEY_F12;
 return KEY_ESC;
}

/*  hardware interrupt  */
void stop_cmd()
{
 repeat = -1;
}
```

オリジナル

```
// DIP-SW setup
const int PinDip1 = 0; // d2
const int PinDip2 = 1; // d3
const int PinDip3 = 2; // d1
const int PinDip4 = 5; // c6 : no use

// Push-Button setup
const int PinDaisen = 7; // e6
const int PinSembok = 6; // d7
const int PinMisato = 4; // d4
const int PinKouiki = 3; // d0
const int PinShift  = 8; // b4

void setup() {

 pinMode( PinDip1, INPUT );
 pinMode( PinDip2, INPUT );
 pinMode( PinDip3, INPUT );
 pinMode( PinDip4, INPUT );

 pinMode( PinDaisen, INPUT );
 pinMode( PinSembok, INPUT );
 pinMode( PinMisato, INPUT );
 pinMode( PinKouiki, INPUT );
 pinMode( PinShift,  INPUT );

}

void loop() {

 // get Push-Button status
 int city = 0, shft = 0;

 if ( digitalRead( PinShift ) == LOW ) {
   shft = 1;
   delay( 80 );
 }

 if      ( digitalRead( PinDaisen ) == LOW ) city = 2120;
 else if ( digitalRead( PinSembok ) == LOW ) city = 2150;
 else if ( digitalRead( PinMisato ) == LOW ) city = 4340;
 else if ( digitalRead( PinKouiki ) == LOW ) city = 8620;

 else if ( shft == 1 ) {  // login windows
   delay( 50 );
   if ( digitalRead( PinShift ) == LOW ) {
     Keyboard.print( "KAIGO" );
     Keyboard.write ( KEY_RETURN );
   }
 }

 // get DIP-SW status
 int pc = 0;
 if ( digitalRead( PinDip1 ) == LOW ) pc += 1;
 if ( digitalRead( PinDip2 ) == LOW ) pc += 2;
 if ( digitalRead( PinDip3 ) == LOW ) pc += 4;
 //if ( digitalRead( PinDip4 ) == LOW ) pc += 8;

 // output !
 if ( city != 0 ) {

   if ( shft == 0 ) { // logout L/P
```

```
    Keyboard.press( KEY_LEFT_SHIFT );
    Keyboard.write( KEY_F12 );
    //Keyboard.print( "test" );
    Keyboard.releaseAll();
    delay( 200 );
  }

  String id = "";
  id += city; id += "000"; id += pc;
  Keyboard.print( id );

  String pw = "000";
  pw += pc;
  Keyboard.print( pw );

  Keyboard.write( KEY_TAB );
  Keyboard.write( KEY_RETURN );
 }

 delay( 50 );
}
```