```
#include <hidboot.h>
#include <usbhub.h>

/* Modified USBHDBOOTKbd Example by Oleg Mazurov (https://www.circuitsathome.com/)
 * This code makes use of the USB host shield designed by Oleg Mazurov and
 * the supporting libaries he wrote for using it.
 */
#ifdef dobogusinclude
#include <spi4teensy3.h>
#endif
#include <SPI.h>

const int Red = A0;
const int Green = A1;
const int Blue = A2;


uint8_t zero = 48;
uint8_t one = 49;
uint8_t two = 50;
uint8_t three = 51;
uint8_t four = 52;
uint8_t five = 53;
uint8_t six = 54;
uint8_t seven = 55;
uint8_t eight = 56;
uint8_t nine = 57;

uint8_t cArr[15];

class KbdRptParser : public KeyboardReportParser
{
    void PrintKey(uint8_t mod, uint8_t key);

  protected:
    void OnControlKeysChanged(uint8_t before, uint8_t after);

    void OnKeyDown  (uint8_t mod, uint8_t key);
//    void OnKeyUp  (uint8_t mod, uint8_t key);
    void OnKeyPressed(uint8_t key);
};

void KbdRptParser::PrintKey(uint8_t m, uint8_t key)
{
  MODIFIERKEYS mod;
  *((uint8_t*)&mod) = m;
  Serial.print((mod.bmLeftCtrl   == 1) ? "C" : " ");
  Serial.print((mod.bmLeftShift  == 1) ? "S" : " ");
  Serial.print((mod.bmLeftAlt    == 1) ? "A" : " ");
  Serial.print((mod.bmLeftGUI    == 1) ? "G" : " ");

  Serial.print(" >");
  PrintHex<uint8_t>(key, 0x80);
  Serial.print("< ");

  Serial.print((mod.bmRightCtrl   == 1) ? "C" : " ");
  Serial.print((mod.bmRightShift  == 1) ? "S" : " ");
  Serial.print((mod.bmRightAlt    == 1) ? "A" : " ");
  Serial.println((mod.bmRightGUI    == 1) ? "G" : " ");
};

void KbdRptParser::OnKeyDown(uint8_t mod, uint8_t key)
{
  uint8_t c = OemToAscii(mod, key);

  if (c)
    OnKeyPressed(c);
  //  Serial.println();
}

void KbdRptParser::OnControlKeysChanged(uint8_t before, uint8_t after) {

  MODIFIERKEYS beforeMod;
  *((uint8_t*)&beforeMod) = before;

  MODIFIERKEYS afterMod;
  *((uint8_t*)&afterMod) = after;

  if (beforeMod.bmLeftCtrl != afterMod.bmLeftCtrl) {
    Serial.println("LeftCtrl changed");
  }
  if (beforeMod.bmLeftShift != afterMod.bmLeftShift) {
    Serial.println("LeftShift changed");
  }
  if (beforeMod.bmLeftAlt != afterMod.bmLeftAlt) {
    Serial.println("LeftAlt changed");
  }
  if (beforeMod.bmLeftGUI != afterMod.bmLeftGUI) {
    Serial.println("LeftGUI changed");
  }
  if (beforeMod.bmRightCtrl != afterMod.bmRightCtrl) {
    Serial.println("RightCtrl changed");
  }
  if (beforeMod.bmRightShift != afterMod.bmRightShift) {
    Serial.println("RightShift changed");
  }
  if (beforeMod.bmRightAlt != afterMod.bmRightAlt) {
    Serial.println("RightAlt changed");
  }
  if (beforeMod.bmRightGUI != afterMod.bmRightGUI) {
    Serial.println("RightGUI changed");
  }

}
/*
void KbdRptParser::OnKeyUp(uint8_t mod, uint8_t key)
{
  Serial.print("UP ");
  PrintKey(mod, key);
}
*/
void KbdRptParser::OnKeyPressed(uint8_t key)
{
  if(key == one)
  {
    Serial.println("*Azure*");
    analogWrite(Red, 0);
    analogWrite(Green, 128);
    analogWrite(Blue, 255);
    delay(10000);
    analogWrite(Red, 0);
    analogWrite(Green, 0);
    analogWrite(Blue, 0);
  }
  else if(key == two)
  {
    Serial.println("*Blue*");
    analogWrite(Red, 0);
    analogWrite(Green, 0);
    analogWrite(Blue, 255);
    delay(10000);
    analogWrite(Red, 0);
    analogWrite(Green, 0);
    analogWrite(Blue, 0);
  }
  else if(key == three)
  {
    Serial.println("*Violet*");
    analogWrite(Red, 192);
    analogWrite(Green, 0);
    analogWrite(Blue, 255);
    delay(10000);
    analogWrite(Red, 0);
    analogWrite(Green, 0);
    analogWrite(Blue, 0);
  }
  else if(key == four)
  {
    Serial.println("*Green*");
    analogWrite(Red, 0);
    analogWrite(Green, 255);
    analogWrite(Blue, 0);
    delay(10000);
    analogWrite(Red, 0);
    analogWrite(Green, 0);
    analogWrite(Blue, 0);
  }
  else if(key == five)
  {
    Serial.println("*Rose*");
    analogWrite(Red, 255);
    analogWrite(Green, 0);
    analogWrite(Blue, 128);
    delay(10000);
    analogWrite(Red, 0);
    analogWrite(Green, 0);
    analogWrite(Blue, 0);
  }
  else if(key == six)
  {
    Serial.println("*Red*");
    analogWrite(Red, 255);
    analogWrite(Green, 0);
    analogWrite(Blue, 0);
    delay(10000);
    analogWrite(Red, 0);
    analogWrite(Green, 0);
    analogWrite(Blue, 0);
  }
```

```
  else if(key == seven)
  {
    Serial.println("*Orange*");
    analogWrite(Red, 255);
    analogWrite(Green, 128);
    analogWrite(Blue, 0);
    delay(10000);
    analogWrite(Red, 0);
    analogWrite(Green, 0);
    analogWrite(Blue, 0);
  }
  else if(key == eight)
  {
    Serial.println("Yellow: ");
    analogWrite(Red, 255);
    analogWrite(Green, 255);
    analogWrite(Blue, 64);
    delay(10000);
    analogWrite(Red, 0);
    analogWrite(Green, 0);
    analogWrite(Blue, 0);
  }
  else if(key == nine)
  {
    Serial.println("*Spring*");
    analogWrite(Red, 0);
    analogWrite(Green, 255);
    analogWrite(Blue, 128);
    delay(10000);
    analogWrite(Red, 0);
    analogWrite(Green, 0);
    analogWrite(Blue, 0);
  }
    Serial.println(key);
};

USB     Usb;
//USBHub     Hub(&Usb);
HIDBoot<USB_HID_PROTOCOL_KEYBOARD>    HidKeyboard(&Usb);

KbdRptParser Prs;

void setup()
{
  pinMode(Red, OUTPUT);
  pinMode(Green, OUTPUT);
  pinMode(Blue, OUTPUT);

  Serial.begin( 9600 );
#if !defined(__MIPSEL__)
  while (!Serial); // Wait for serial port to connect - used on Leonardo, Teensy
and other boards with built-in USB CDC serial connection
#endif
  Serial.println("Start");

  if (Usb.Init() == -1)
    Serial.println("OSC did not start.");

  delay( 200 );

  HidKeyboard.SetReportParser(0, &Prs);

}

void loop()
{
  Usb.Task();
}
```