```
#include <QueueList.h>
#include "qrcode.h"
#include "Wire.h"
#include <Adafruit_GFX.h> // Hardware-specific library
#include <MCUFRIEND_kbv.h>
#include <avr/wdt.h>
```

Import ทั้งหมด

queueList สร้าง queue

qrcode สร้าง qrcode ตามสั่ง

Wire เรียกใช้ i2c

Adafruit กับ Mcufriend ไว้คุยกับ lcd

arv/wdt ใช้เรียก watchdog timer

```
#define BLACK   0x0000
#define BLUE    0x001F
#define RED     0xF800
#define GREEN   0x07E0
#define CYAN    0x07FF
#define MAGENTA 0xF81F
#define YELLOW  0xFFE0
#define WHITE   0xFFFF
```

Define สีเพื่อกำหนดใช้ใน lcd

‘

```
enum Protocol {
  HEADER,
  COMMAND,
  RESET,
  O_C,
  SHOWTEXT,
  SHOWQR,
  SHOWTITLE,
  END
};

enum State {
  QR,
  TEXT
};
```

Protocol เอาไว้ทำ state รับคำสั่ง
State เอาไว้เก็บค่าว่า จะโชว์ text หรือ qr ที่จอ

```
MCUFRIEND_kbv tft;
uint16_t g_identifier;
QueueList <uint8_t> queue;
Protocol protocolUart;
State state;
uint8_t command;
uint8_t qrSize;
uint8_t textSize;
uint8_t titleSize;
uint8_t qrData[128];
uint8_t textData[128];
uint8_t titleData[128];
uint8_t inputByte;
bool o_c;
```

mcufriend_kbv ไว้ init จอ
g_identifier ไว้ หา id ของจอ
Command เก็บค่าคำสั่งไว้ใช้ตอน process
Size เก็บขนาดของแต่ละอย่าง
Data ข้อมูลของแต่ละอย่าง
o_c เช็คว่าจอเปิดหรือปิด

```
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);

  g_identifier = tft.readID();

  Serial.print("ID = 0x");
  Serial.println(g_identifier, HEX);

  tft.begin(g_identifier);
  tft.fillScreen(WHITE); //fillScreen(uint16_t t);
  tft.setTextSize(5);
  tft.setTextColor(BLACK);
  tft.setCursor(0, 0);
  tft.setRotation(0);

  Serial.println(tft.width()); //int16_t width(void);
  Serial.println(tft.height()); //int16_t height(void);


  Wire.begin(0x71);
  Wire.onReceive(receiveEvent); // register event
  o_c = true;

  wdt_enable(WDTO_1S);
}
```

```
void receiveEvent(int howMany) {

  while (0 < Wire.available()) { // loop through all
    uint8_t c = Wire.read(); // receive byte as a character
    queue.push(c);
    Serial.print(c);         // print the character
  }

  wdt_reset();
  getProtocol();
}
```

ฟังชั่นรับข้อมูล i2c จาก master แล้วจัดเก็บไว้ใน queue จากนั้นสั่ง get protocol

```
(* resetFunc) (void) = 0; //declare reset function @ address 0
```

ฟังชั่นไว้ reset บอร์ด

```
void getProtocol() {
  wdt_reset();
  inputByte = queue.pop();
  uint8_t dataSize;
  switch (protocolUart) {
    case HEADER :
      if (inputByte == 0x02) {
        protocolUart = COMMAND;
      }
      break;
    case COMMAND :
      inputByte = queue.pop();
      if (inputByte == 0x21) {
        protocolUart = RESET;
      }
      else if ( inputByte == 0x22) {
        protocolUart = O_C;
      }
      else if ( inputByte == 0x23) {
        protocolUart = SHOWQR;
      }
      else if (inputByte == 0x24) {
        protocolUart = SHOWTEXT;
      }

    case RESET :
      dataSize = queue.pop();
      inputByte = queue.pop();
      if ( dataSize == 0x01) {
        if (inputByte == 0x00) {
          command = 0x01;
          protocolUart = END;
        }

      } else {
        protocolUart = HEADER;
      }
      break;
```

inputByte แรกเช็คว่าตัวแรก 0x02 เข้า command

พอเข้า command ดูว่าสั่งะไร

```
case O_C :
  dataSize = queue.pop();
  inputByte = queue.pop();
  if ( dataSize == 0x01) {
    if (inputByte == 0x00) {
      command = 0x11;
      protocolUart = END;
    }
    else if (inputByte == 0x01) {
      command = 0x12;
      protocolUart = END;
    }
  } else {
    protocolUart = HEADER;
  }
  break;

case SHOWQR :
  dataSize = queue.pop();
  if ( dataSize <= 0x00 && dataSize >= 0xFF) {
    qrSize = dataSize;
    command = 0x03;
    for (int i = 0 ; i < dataSize ; i++) {
      qrData[i] = queue.pop();
    }
    protocolUart = END;
  } else {
    protocolUart = HEADER;
  }
  break;

case SHOWTEXT :
  dataSize = queue.pop();
  if ( dataSize <= 0x00 && dataSize >= 0xFF) {
    textSize = dataSize;
    command = 0x04;
    for (int i = 0 ; i < dataSize ; i++) {
      textData[i] = queue.pop();
    }
```

o_c ดูเพื่อหาว่าเปิดหรือปิด

showQr เก็บขนาดใน dataSize วน
ลูปไวาใน array ที่ประกาศไว้ตอนแรก

showText กับ showTitle คอนเซปเดียวกัน

```
void processProtocol() {
  wdt_reset();
  if (command == 0x01) {
    resetFunc();
  }
  else if (command == 0x11) {
    //off
    offDisplay();
  }
  else if (command == 0x12) {
    //on
    onDisplay();
  }
  else if (command == 0x03) {
    //show qr
    state = QR;
    tft.reset();
    tft.fillScreen(WHITE);
    printHeader();
    printQR();
  }
  else if (command == 0x04) {
    //show text
    state = TEXT;
    tft.reset();
    tft.fillScreen(WHITE);
    printHeader();
    printText();
  }

  else if (command == 0x05) {
    //set title
    tft.reset();
    tft.fillScreen(WHITE);
    printHeader();
    if (state == QR) {
      printQR();
    }
    else {
```

เรียกดูค่าใน command ไม่ว่าจะสั่งอัพเดตข้อมูลอะไรก็ตามให้ล้างทุกอย่างบนจอแล้วสั่งปริ้นใหม่โดยเรียกข้อมูลจากที่เซฟไว้ในตัวแปรต่างๆ

```
void onDisplay() {
  wdt_reset();
  if ( o_c == false) {
    tft.reset();
    tft.fillScreen(WHITE);
    printHeader();
    if (state == QR) {
      printQR();
    }
    else {
      printText();
    }
    o_c = true;
  }
}

void offDisplay() {
  wdt_reset();
  tft.reset();
  tft.fillScreen(BLACK);
  o_c = false;
}

void printHeader() {
  wdt_reset();
  tft.setTextSize(5);
  tft.setCursor(1, 0);
  for (int i = 0; i < titleSize; i++) {
    tft.write(titleData[i]);
  }
}

void printQR() {
  wdt_reset();
  QRCode qrcode;
  uint8_t qrcodeData[qrcode_getBufferSize(6)];
  qrcode_initBytes(&qrcode, qrcodeData, 6, 3, qrData, qrSize);
  for (uint8_t y = 0; y < qrcode.size; y++) {
```

ทุกครั้งที่สั่งเปิดปิดจะตั้ง o_c ทุกครั้งและถ้าเปิดอยู่จะไม่สามารถสั่งเปิดซ้ำได้

Limit ของแต่ละส่วน

Header : สามารถใส่ได้มากสุด 13 ตัวอักษรต่อแถว
Text : สามารถใส่ได้มากสุด 17 ตัวอักษรต่อแถว
QR code : สามารถ generate ได้มากสุดที่ 134 ไบต์
*i2c ถ้าส่งเร็วเกินไปจอภาพจะไม่สามารถประมวณผล qr code ได้ทันต้องใส่ delay เผื่อไว้ระดับหนึ่ง