```
//MCU ATMEGA8 throught MiniCore by https://github.com/MCUdude/MiniCore/

/*
say Ahoj on start - it's mean hello in Czech

T1    DISPLAY    T2
S1               S2

long press T1 - set time for S1
long press T2 - set time for S2
long press T1+T2 together - reset coffee counter
press S1, S2 to run grinder
*/

#include <EEPROM.h>

#define Q1 A0
#define Q2 A1
#define Q3 A2
#define Q4 A5
#define Q5 A3
#define Q6 A4

#define SW1 SS
#define SW2 MOSI
#define SW3 MISO
#define SW4 SCK

#define PORT_C 2
#define PORT_D 3

#define DISPTIME1 3000  //us for 1 digit
#define DISPTIME0 10    //us for 1 digit

#define ADRT1 10  //time1  x100ms 10-255
#define ADRT2 20  //time2  x100ms 10-255
#define ADRCC 30  //counter 0-9999   adress 30,31
```

```
//time for accept pressed button SW#
#define TIME_SW1 1000
#define TIME_SW2 1000
#define TIME_SW3 10
#define TIME_SW4 10
#define TIME_SW5 3000  //SW1+SW2 for reset coffee counter
#define TIME_SW6 10    //SW1 -
#define TIME_SW7 10    //SW1 +
#define TIME_SW8 3000  //noKey

int runMs, countdown, counter = 0;
int time1, time2;  //x100ms
int oldtime1, oldtime2;


byte sw[8] = { 0, 0, 0, 0, 0, 0, 0, 0 }, firstClick[8] = { 0, 0, 0, 0,
0, 0, 0, 0 };
unsigned long timesw[8];
unsigned long runtime, cdtime;

byte status = 0;  //0: ahoj 1: disp counter; 2: run T1; 3: run T2; 4:
set T1; 5: set T2

void testSW() {  //set sw[#]=true if pressed for more than TIME_SWx ms
  sw[0] = false;
  if (!digitalRead(SW1) && digitalRead(SW2) && firstClick[0]) {
    firstClick[0] = false;
    timesw[0] = millis();
  }
  if (digitalRead(SW1)) firstClick[0] = true;
  if ((!digitalRead(SW1)) && (digitalRead(SW2)) && (!firstClick[0]) &&
(millis() - timesw[0] > TIME_SW1)) sw[0] = true;

  sw[1] = false;
  if (!digitalRead(SW2) && digitalRead(SW1) && firstClick[1]) {
    firstClick[1] = false;
    timesw[1] = millis();
  }
```

```
  if (digitalRead(SW2)) firstClick[1] = true;
  if ((!digitalRead(SW2)) && (digitalRead(SW1)) && (!firstClick[1]) &&
(millis() - timesw[1] > TIME_SW2)) sw[1] = true;

  sw[2] = false;
  if (!digitalRead(SW3) && firstClick[2]) {
    firstClick[2] = false;
    timesw[2] = millis();
  }
  if (digitalRead(SW3)) firstClick[2] = true;
  if ((!digitalRead(SW3)) && (!firstClick[2]) && (millis() - timesw[2]
> TIME_SW3)) sw[2] = true;

  sw[3] = false;
  if (!digitalRead(SW4) && firstClick[3]) {
    firstClick[3] = false;
    timesw[3] = millis();
  }
  if (digitalRead(SW4)) firstClick[3] = true;
  if ((!digitalRead(SW4)) && (!firstClick[3]) && (millis() - timesw[3]
> TIME_SW4)) sw[3] = true;

  sw[4] = false;
  if (!digitalRead(SW1) && !digitalRead(SW2) && firstClick[4]) {
    firstClick[4] = false;
    timesw[4] = millis();
  }
  if (digitalRead(SW1) && digitalRead(SW2)) firstClick[4] = true;
  if ((!digitalRead(SW1)) && (!digitalRead(SW2)) && (!firstClick[4]) &&
(millis() - timesw[4] > TIME_SW5)) sw[4] = true;

  sw[5] = false;
  if (!digitalRead(SW1) && digitalRead(SW2) && firstClick[5]) {
    firstClick[5] = false;
    timesw[5] = millis();
  }
  if (digitalRead(SW1)) firstClick[5] = true;
```

```
  if ((!digitalRead(SW1)) && (digitalRead(SW2)) && (!firstClick[5]) &&
(millis() - timesw[5] > TIME_SW6)) sw[5] = true;

  sw[6] = false;
  if (!digitalRead(SW2) && digitalRead(SW1) && firstClick[6]) {
    firstClick[6] = false;
    timesw[6] = millis();
  }
  if (digitalRead(SW2)) firstClick[6] = true;
  if ((!digitalRead(SW2)) && (digitalRead(SW1)) && (!firstClick[6]) &&
(millis() - timesw[6] > TIME_SW7)) sw[6] = true;

  sw[7] = false;
  if ((!sw[5]) && (!sw[6]) && (!sw[2]) && (!sw[3]) && (firstClick[7]))
{
    firstClick[7] = false;
    timesw[7] = millis();
  }
  if ((sw[5]) || (sw[6]) || (sw[2]) || (sw[3])) firstClick[7] = true;
  if ((!sw[5]) && (!sw[6]) && (!sw[2]) && (!sw[3]) && (!firstClick[7])
&& (millis() - timesw[7] > TIME_SW8)) sw[7] = true;
}

void setnum(byte x) {
  const byte seg[22] = {
    /* my PINout
       LED segments on digital pins
       7 6 5 4 3 2 1

       my pinout
       ---A---      --7--
       F     B     2     6
       ---G---      --1--
       E     C     3     5
       ---D---      --4--


    */
```

```
    B1000000,  //0
    B1111001,  //1
    B0100100,  //2
    B0110000,  //3
    B0011001,  //4
    B0010010,  //5
    B0000010,  //6
    B1111000,  //7
    B0000000,  //8
    B0010000,  //9
    B1111111,  //10 off
    B0111111,  //11 -
    B0101111,  //12 r
    B1100011,  //13 u
    B0101011,  //14 n
    B0000111,  //15 t
    B0001000,  //16 A
    B0001011,  //17 h
    B0100011,  //18 o
    B1110001,  //19 J
    B0000110,  //20 E
    B1000001,  //21 U
  };
  portWrite(PORT_D, seg[x]);
}

void disp(byte adr, byte num) {
  //adr led disp 0-3
  digitalWrite(Q1, HIGH);
  digitalWrite(Q2, HIGH);
  digitalWrite(Q3, HIGH);
  digitalWrite(Q5, HIGH);
  delayMicroseconds(DISPTIME0);
  switch (adr) {
    case 0:
      digitalWrite(Q1, LOW);
      break;
    case 1:
```

```
      digitalWrite(Q2, LOW);
      break;
    case 2:
      digitalWrite(Q3, LOW);
      break;
    case 3:
      digitalWrite(Q5, LOW);
      break;
  }
  setnum(num);
  delayMicroseconds(DISPTIME1);
}

void writeCounter() {  //to EEPROM
  byte byte1 = counter;
  byte byte2 = counter >> 8;
  EEPROM.write(ADRCC, byte1);
  EEPROM.write(ADRCC + 1, byte2);
}

void readCounter() {  //from EEPROM
  byte byte1 = EEPROM.read(ADRCC);
  byte byte2 = EEPROM.read(ADRCC + 1);
  counter = byte1 + (byte2 << 8);
}

void plusCounter(byte x) {
  counter = counter + x;
  if (counter > 9999) counter = 0;
  writeCounter();
}

void dispNum(int number) {
  int x = number;
  int t = x / 1000;
  x = x - t * 1000;
  int s = x / 100;
  x = x - s * 100;
```

```
  int d = x / 10;
  int j = x % 10;
  disp(0, t);
  disp(1, s);
  disp(2, d);
  /*
    if (number > 999) disp(0, t);
    if (number > 99) disp(1, s);
    if (number > 9) disp(2, d);
  */
  disp(3, j);
}

void dispRun1() {
  disp(0, 12);  //r
  disp(1, 13);  //u
  disp(2, 14);  //n
  disp(3, 1);   //1
}

void dispRun2() {
  disp(0, 12);  //r
  disp(1, 13);  //u
  disp(2, 14);  //n
  disp(3, 2);   //1
}


void dispT1() {
  disp(0, 15);  //t
  disp(1, 1);   //1
  disp(2, 10);  //off
  disp(3, 10);  //off
}

void dispT2() {
  disp(0, 15);  //t
  disp(1, 2);   //2
```

```
  disp(2, 10);  //off
  disp(3, 10);  //off
}

void dispAhoj() {
  disp(0, 16);  //A
  disp(1, 17);  //H
  disp(2, 18);  //0
  disp(3, 19);  //J
}

void dispReset() {
  disp(0, 11);  //-
  disp(1, 11);  //-
  disp(2, 11);  //-
  disp(3, 11);  //-
}

void dispSave() {
  for (int j = 1; j <= 10; j++) {
    for (int i = 1; i <= 10; i++) {
      disp(0, 5);   //S
      disp(1, 16);  //A
      disp(2, 21);  //V
      disp(3, 20);  //E
    }
    for (int i = 1; i <= 5; i++) {
      disp(0, 10);  //off
      disp(1, 10);  //off
      disp(2, 10);  //off
      disp(3, 10);  //off
    }
  }
}

void doDisp(int x) {
  for (int i = 1; i <= x; i++) {
    switch (status) {
```

```
      case 0:  //start only
        dispAhoj();
        break;
      case 1:
        dispNum(counter);
        break;
      case 2:
        dispRun1();
        break;
      case 3:
        dispRun2();
        break;
      case 4:
        dispT1();
        break;
      case 5:
        dispT2();
        break;
      case 6:
        dispReset();
        break;
      case 7:  //set time1
        dispNum(time1);
        break;
      case 8:  //set time2
        dispNum(time2);
        break;
      case 9:  //countdown
        dispNum(countdown);
        break;
    }
    testSW();
  }
}

void setup() {
  pinMode(Q1, OUTPUT);
  pinMode(Q2, OUTPUT);
```

```
  pinMode(Q3, OUTPUT);
  pinMode(Q5, OUTPUT);
  pinMode(Q6, OUTPUT);

  pinMode(PD0, OUTPUT);  //A
  pinMode(PD1, OUTPUT);  //B
  pinMode(PD2, OUTPUT);  //C
  pinMode(PD3, OUTPUT);  //D
  pinMode(PD4, OUTPUT);  //E
  pinMode(PD5, OUTPUT);  //F
  pinMode(PD6, OUTPUT);  //G

  pinMode(SW1, INPUT_PULLUP);
  pinMode(SW2, INPUT_PULLUP);
  pinMode(SW3, INPUT_PULLUP);
  pinMode(SW4, INPUT_PULLUP);

  doDisp(160);
  status = 1;  //disp counter
  time1 = EEPROM.read(ADRT1);
  time2 = EEPROM.read(ADRT2);
  readCounter();  //from EEPROM
}

void loop() {
  doDisp(5);

  if ((sw[0]) && (status == 1)) {  //set time1 from disp counter only
    status = 4;
    doDisp(100);
    oldtime1 = time1;
    status = 7;
  }

  if ((sw[5]) && (status == 7)) {  //minus
    time1--;
    if (time1 < 10) time1 = 10;
  }
```

```
if ((sw[6]) && (status == 7)) {  //plus
  time1++;
  if (time1 > 255) time1 = 255;
}

if ((sw[7]) && (status == 7)) {  //no key if setting time
  if (time1 != oldtime1) {       //save new time1
    EEPROM.write(ADRT1, time1);
    dispSave();
  }
  status = 1;
}

if ((sw[1]) && (status == 1)) {  //set time2 from disp counter only
  status = 5;
  doDisp(100);
  oldtime2 = time2;
  status = 8;
}

if ((sw[5]) && (status == 8)) {  //minus
  time2--;
  if (time2 < 10) time2 = 10;
}

if ((sw[6]) && (status == 8)) {  //plus
  time2++;
  if (time2 > 255) time2 = 255;
}

if ((sw[7]) && (status == 8)) {  //no key if setting time1
  if (time2 != oldtime2) {       //save new time2
    EEPROM.write(ADRT2, time2);
    dispSave();
  }
  status = 1;
}
```

```
if (sw[2]) {  //run1
  status = 2;
  digitalWrite(Q6, HIGH);
  runtime = millis();
  cdtime = runtime;
  runMs = 100 * time1 + 100;
  countdown = runMs / 1000;
  while (!(millis() - runtime > runMs)) {
    doDisp(10);
    if ((status != 9) && (millis() - runtime > 1000)) status = 9;
    if (millis() - cdtime > 1000) {
      countdown = countdown - 1;
      if (countdown < 0) countdown = 0;
      cdtime = millis();
    }
  }
  digitalWrite(Q6, LOW);
  plusCounter(1);  //coffeeCounter++
  status = 1;      //return to disp counter
}

if (sw[3]) {  //run2
  status = 3;
  digitalWrite(Q6, HIGH);
  runtime = millis();
  cdtime = runtime;
  runMs = 100 * time2 + 100;
  countdown = runMs / 1000;
  while (!(millis() - runtime > runMs)) {
    doDisp(10);
    if ((status != 9) && (millis() - runtime > 1000)) status = 9;
    if (millis() - cdtime > 1000) {
      countdown = countdown - 1;
      if (countdown < 0) countdown = 0;
      cdtime = millis();
    }
  }
```

```
    digitalWrite(Q6, LOW);
    plusCounter(2);  //coffeeCounter++
    status = 1;      //return do disp counter
  }

  if (sw[4]) {  //reset counter
    status = 6;
    doDisp(100);
    for (int i = counter; i >= 0; i = i - 10) {  //funny countdown
      counter = i;
      status = 1;
      doDisp(1);
    }
    counter = 0;
    writeCounter();
  }
}
```