/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Target MCU: ATTiny13, 9.6Mhz Internal Clcok

Name : CLKinternal\_MenuAlarm\_NoMillis\_SettableAlarm.ino

Author : Insoo Kim

Date : March 20, 2015

Notes : Set alarm by pressing button counts

Power on Default operation:

Press the button

once for 3 min alarm,

twice for 5 min,

3 times for 15 min

To set temporary alarm period and change the "once" button temporarily,

press the button 4 times, like seleting menu

after blinking LED 4 times to confirm your menu selection

press your temporary alarm period which will be assinged to the

"once" button.

Your temporary setting is maintained only before power recycling

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

boolean alarmEnable = false;

boolean start = false;

boolean BLINK\_NOTICED = false;

#define BUTTON\_MENU 0

#define BUTTON\_TEMP\_ALARM\_NUM 1

//Duration between numbers

//#define DURATION 2900 // for 170 pin bread b'd

//#define buzzPin 4 // for 170 pin bread b'd

//#define startPin 3 // for 170 pin bread b'd

#define DURATION 370 // for 2\*8 perf b'd

//#define menuSelCompleteINTERVAL (DURATION\*2)

#define menuSelCompleteINTERVAL 8

#define shortBuzz 3 // buzzing 3 times

#define longBuzz 10 // buzzing 10 times

#define buzzPin 3 // for 2\*8 perf b'd

#define startPin 4 // for 2\*8 perf b'd

#define ledPin 0 // for 2\*8 perf b'd

byte clockCnt;

byte secCnt;

byte minCnt;

byte alarm[3] = {3, 5, 15};

byte menuCnt=0, tempAlarmCnt=0;

byte prevLoop=0, curLoop=0, lapse=0;

byte loopCnt=0;

//-------- FUNCTION PROTOTYPES

// Arduino Sketch C doesn't need to declare function prototypes

// But to conform with ANSI C, here i follow the standard C rules.

void startClock(byte );

void countButton(byte);

void blinkLED(byte );

void buzz(byte);

void chkAlarm(byte );

//-------- SETUP

void setup() {

pinMode(startPin, INPUT);

pinMode(buzzPin, OUTPUT);

pinMode(ledPin, OUTPUT);

}//setup

//-------- LOOP

void loop()

{

loopCnt++;

if (menuCnt <= 3)

countButton(BUTTON\_MENU);

else if (menuCnt == 4)

countButton(BUTTON\_TEMP\_ALARM\_NUM);

curLoop = loopCnt;

lapse = curLoop - prevLoop;

if (lapse > menuSelCompleteINTERVAL)

{

if (menuCnt != 0)

{

loopCnt = 0;

if (!BLINK\_NOTICED)

{

blinkLED(menuCnt);

}//if (!BLINK\_NOTICED)

switch (menuCnt)

{

case 1:

alarmEnable = true;

startClock(alarm[0]);

break;

case 2:

alarmEnable = true;

startClock(alarm[1]);

break;

case 3:

alarmEnable = true;

startClock(alarm[2]);

break;

}//switch (menuCnt)

}//if (menuCnt != 0)

//when menuCnt == 4, buttonCount function counts "tempAlarmCnt"

if (tempAlarmCnt != 0)

{

loopCnt = 0;

if (!BLINK\_NOTICED)

{

blinkLED(menuCnt);

}//if (!BLINK\_NOTICED)

//DONE\_incUnit = true;

blinkLED(tempAlarmCnt);

alarm[0] = tempAlarmCnt;

tempAlarmCnt = 0;

menuCnt = 1;

}//if (incUnitCnt != 0)

}//if (lapse > menuSelCompleteINTERVAL)

if (!start)

{

//delay should be short enough to catch button press by user

delay(DURATION/4);

}

}//loop

//-----------------------------------

void startClock(byte alarmMin)

{

start = true;

clockCnt=0;

secCnt=0;

minCnt=0;

while (start)

{

clockCnt++;

if (clockCnt % 2 == 0)

secCnt++;

//check minute

if (secCnt == 60)

{

minCnt++;

clockCnt = 0;

blinkLED(menuCnt);

//blinkLED(menuCnt) routine consumes around 1 sec, so we need to complement the loss

secCnt = 1;

}

//========== check Alarm enable status

if (alarmEnable == true)

{

//digitalWrite(ledPin, HIGH);

chkAlarm(alarmMin);

}

else

{

//digitalWrite(ledPin, LOW);

start = false;

}

delay(DURATION); // delay in between reads for stability

//DelayNoBlock(DURATION);

}//while (start)

}//startClock

//-----------------------------------

void countButton(byte cate)

{

//if pressed, LOW

if (digitalRead(startPin) == LOW)

{

delay(200); // for debounce

switch (cate)

{

case BUTTON\_MENU:

menuCnt++;

break;

case BUTTON\_TEMP\_ALARM\_NUM:

tempAlarmCnt++;

break;

} //switch (cate)

prevLoop = loopCnt;

}//if (digitalRead(startPin) == LOW)

}//countButton

//-----------------------------------

void blinkLED(byte num)

{

byte i;

for (i=0; i<num; i++)

{

digitalWrite(ledPin, HIGH);

delay(DURATION/3);

digitalWrite(ledPin, LOW);

delay(DURATION/3);

}

BLINK\_NOTICED = true;

}//blinkLED

//-----------------------------------

void buzz(byte times)

{

const byte buzzInterval = 200;

byte i;

for (i=0; i<times; i++)

{

digitalWrite(buzzPin, HIGH);

delay(buzzInterval);

digitalWrite(buzzPin, LOW);

delay(buzzInterval);

}

}//buzz

//-----------------------------------

void chkAlarm(byte num)

{

//if the current minute has reached to alarm set

if(num == minCnt)

{

//buzzing

buzz(shortBuzz);

//disable alarm setting

alarmEnable = false;

//turn off the set alarm LED

digitalWrite(ledPin, LOW);

//reset menu selection count

menuCnt=0;

//prevMS = millis();

prevLoop = loopCnt;

start = false;

BLINK\_NOTICED = false;

}

}//chkAlarm