UDK 004.421.4

V.M. Zayats, V.R. Kolosov
National University "Lviv Politechnic", Lviv
General ecology and ecology-informational systems department

ALGORITHMS AND SOFTWARE ENVIRONMENT OF TIME FORMAT NUMBERS ARITHMETIC FOR THE AVR FAMILY MICROCONTROLLER

Запропоновано способи виконання арифметичних операцій додавання та віднімання чисел, поданих у форматі часу. Розроблено алгоритми та програмні засоби на алгоритмічній мові С для мікроконтролера сімейства AVR.

***Ключові слова*: арифметичні операції, формат часу, режим реального часу, алгоритм, програма.**

The method of implementation of operations of addition and subtraction of numbers, presented in the time-format is considered. The algorithm and program is developed by the C-language for microcontrollers of AVR family.

***Keywords:* arithmetic operations, time format, real-time algorithm, program.**

Introduction

Digital devices usually execute operations on transformation of information in the binary scale of notation. But often there is a necessity to process information in the time- format (for example, readed from real-time clocks). The most widespread operations are operations of addition and subtraction of numbers.

Authors of similar operations or suggest to take advantage of existent additions (for example, Microsoft Excel [1]), or application of variables of type of string [2], or the use in quality of intermediate BCD-format [3].

Purpose of work

The purpose of work is development of algorithms, that realization of programmatic facilities for implementation of arithmetic operations of addition and subtraction of numbers, given in the time-format (gg:hh:ss) and to confirm expedience of his appendix to microcontroller of AVR-family [4-7].

Review of literary sources. AVR-microcontrollers of are development and product of firm Atmel. It is family of universal, 8-bit microcontrollers of RISC-architecture (the program and information are in different address spaces) with different built-in peripheral units (ADC, receiver-transmitter, TWI-module, counters, SPI and a number of other devices depending on a model). Made on technology of 0,35 μm., work with a clock rate from 16 MHz. A firm ATMEL produces families of 8-bit microcontrollers of two types: tiny and mega. The microcontrollers of tiny have FLESH-ROM for 1 and 2 KB in a corps on 8-20 pins, and microcontrollers of mega accordingly: FLESH-ROM of 8-128 KB in a corps on by 28-64 pins, can work for a voltage feed a 2-6 volt. Is possible to switch them in the modes of economy consumption of energy by a programmatic way.

At the end of 1996 year the first experimental microcontroller of AT90S1200 was produced, and in the second half of 1997 the Atmel corporation began the mass production of new family of microcontrollers, carrying out here them publicity and technical support.

The basic versions of controllers are such:

- AT (mega / tiny) xxx is a base version.
- ATxxxL are versions of controllers which work at reduced (Low) feed voltage (2,7 V).
- ATxxxV are versions of controllers which work on low feed voltage (1,8 V).
- ATxxxP is low consumed versions (to 100 nA in the Power-down mode), picopower technology (announced in July 2007) [4] is applied, by the number of pins and functional capabilities compatible with previous versions.

In [5] functioning principles, architecture feature, and programming technique of Atmel AVR microcontrollers are expounded. Ready methods of programming the basic functions of modern microelectronic apparatus are showed: from a reaction on pressure of the button or construction of dynamic indication to difficult protocols of record of data in external memory or features of connecting of real time clock. The special attention gives the data exchange of microelectronic devices and the personal computer,

examples of the programs are given. The features of AVR modern models and concomitant microcircuits of the last years of issue are taken into account in a book.

All design phases of microcontrollers devices are showed in [6]. The special attention is spared connection of the offered schematics with software of designed device. The electric charts of devices are offered to the head in each - controllers on the base of AVR-microcontrollers, and also a few programs which determine functioning of these controllers. Basic applied applications microcontrollers of this class.

Next to base AVR-microcontrollers exist other technologies of construction of these devices develop successfully. So exhaustive description of base series of AVR-microcontrollers family from a company Atmel, built on the base of progressive RISC-architecture with application of programmable flesh-EPROM memory is given in [7]. In addition, programming of microcontrollers of this series is thoroughly examined in assembly language, and also environment of adjusting of AVR-Studio and hardware-software set of STK200.

The main part

Time-format arithmetic complicated that they do not answer requirements which are produced to the classic numerical notations.

Thus, for presentation of time-sizes utilized numbers 0 ... 9, as well as in decimal numerical notation, at the same time exist limit on the maximal number of seconds, minutes and hours. So, seconds and minutes change scope from 0 to 59, and hours - from 0 to 23 (in the 24-hours time-format).

For example, at addition 40 and 30 seconds nonmetering these limitations result there will be 70 seconds, but with a glance - 1 minute 10 seconds.

One of variants of decision of problem consists in the decimal correction of result of implementation of operation and by turn correction of value of seconds, minutes and hours.

Second is in converting of number from the seconds-minutes-housr-format to the seconds-format, implementation of operation and reverse transformation.

Describe each of possible approaches in algorithmic representation.

First algorithm.

In an order not to execute a decimal correction both after addition and after subtraction which considerably complicates an algorithm (practically in two times), data translation one of operands in a complement code which will allow to replace the subtraction-operation of addition-operation (fig. 1).

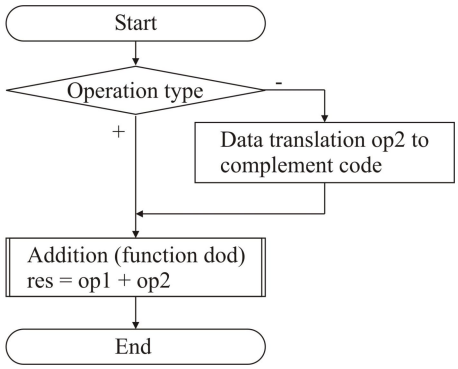


Fig. 1. Algorithmic diagram of addition/subtraction of time-format numbers.

Addition procedure of two numbers with the next decimal correction of result is showed on fig. 2.

A negative number is presented in a complement code. The decimal correction of result of addition is executed taking into account circumstance that the numbers of units of seconds, minutes and hours (even digits of number), change from "0" to "9", and numbers of ten of seconds and minutes - from "0" to "5". The amount of ten of hours in this algorithm is unlimited.

Second algorithm.

For converting of number in seconds can be utillized the widespread method of weighting coefficients, essence of which consists in addition of product each of digits of number and its weight. Weight of digits is in seconds for a number, presented in the time-format, represented in a table 1.

Table 1.

Weight	36000	3600	600	60	10	1
Digit	Ten of hours	Hours	Ten of minutes	Minutes	Ten of seconds	Seconds

Then

$$sek = dg*36000 + g*3600 + dh*600 + h*60 + ds*10 + s, \quad (1)$$

where sek – result in seconds;
dg– ten of hours number;
g – hours number;
dh – ten of minutes number;
h – minutes number;
ds – ten of seconds number;
s – seconds number.

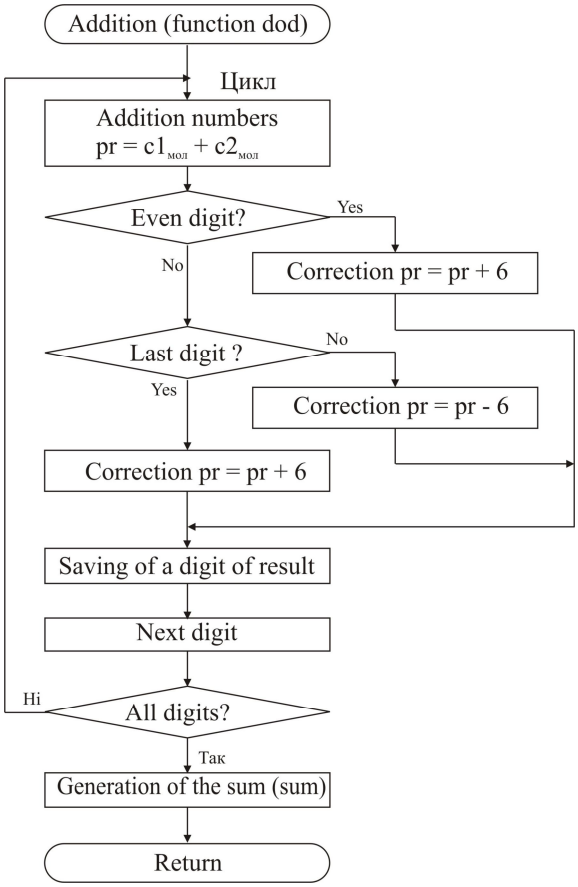


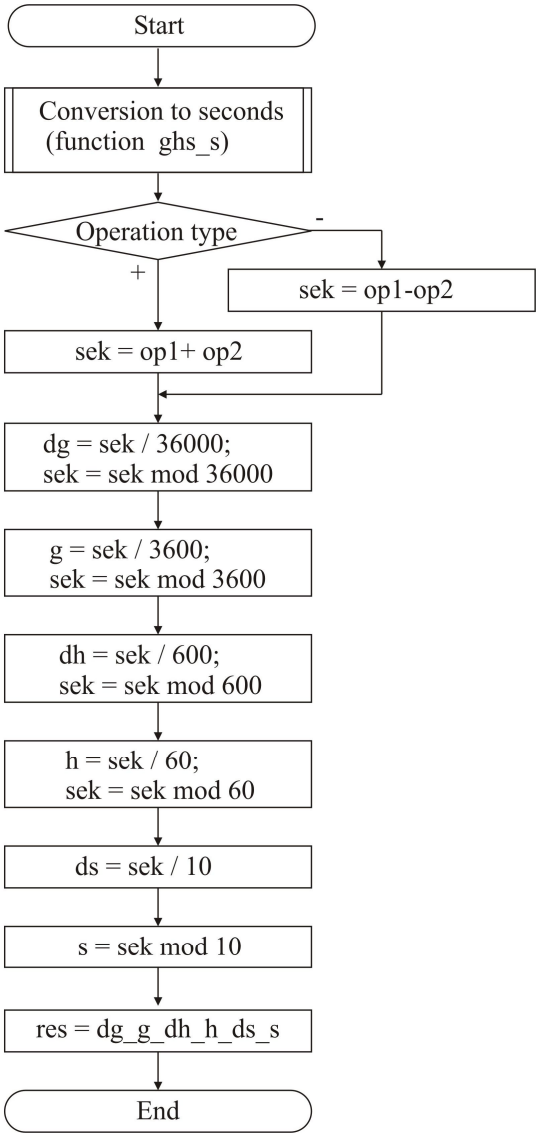
Fig. 2. Algorithmic diagram of addition of two numbers with the next decimal correction.

For reverse transformation comfortably to utillize the method of successive calculation of numbers: ten of hours, hours, ten of minutes, minutes, ten of seconds, seconds. A calculation is executed by next algorithm:

- a number in seconds is divided by weight of number of ten of hours. A quotient gives the number of ten of hours;
- a remain is divided by weight of number of hours. A quotient gives the number of hours;
- a remain is divided by weight of number of ten of minutes. A quotient gives the number of ten of minutes;
- a remain is divided by weight of number of minutes. A quotient gives the number of minutes;

- a remain is divided by weight of number of ten of seconds. A quotient gives the number of ten of seconds, a remain gives the number of seconds.

Algorithmic diagram is represented on fig. 3.



Puc. 3. Algorithmic diagram of addition/subtraction of time-format numbers (representation through the seconds).

Algorithmic diagram of converting of number from the seconds-minutes-housr-format to the seconds-format represented on fig. 4.

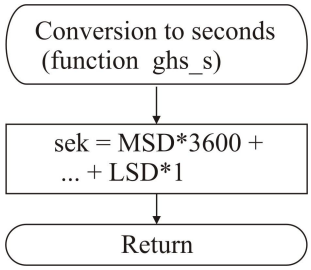


Fig. 4. Algorithmic diagram of converting of number from the seconds-minutes-housr-format to the seconds-format.

Comfortably to present both programs as parametrized function in which at its call passed type of operation (addition or subtraction) and operands value.

Returned value, and will be the sought quantity.

Unsealing of the program, realized by the algorithm 1, is showed on Listing 1.

Listing 1. The program of implementation addition/subtraction operations of time-format numbers.

```
#include <avr/io.h>
#define u8          unsigned char
#define u32         unsigned long
volatile u32 a,b,c,d;                    // time-format numbers (hours-minutes-seconds)
//---- data transformation of hours, minutes and seconds to seconds subroutine ---
u32 ghs_s(u32 ghs){ volatile u32 sek;  // seconds
sek=(((ghs>>20)&0x0F)*36000)+(((ghs>>16)&0x0F)*3600)+(((ghs>>12)&0x0F)*600)+(((ghs>>8)&0x0F)*60)+(((ghs>>4)&0x0F
)*10)+(ghs&0x0F);
 return sek;
}
//----------- time-format numbers arithmetic operation subroutine ----------------
u32 ar_op(u8 op, u32 op1, u32 op2){
 volatile u8 s,ds,h,dh,g,dg;                    //digits of the source number
 volatile u32 sek,res;                          //result

 if(op==0)          sek=ghs_s(op1)+ghs_s(op2);          // addition
 else               sek=ghs_s(op1)-ghs_s(op2); // subtraction
dg=sek/36000; sek%=36000;                       //calculation of ten of hours
 g=sek/3600; sek%=3600;                         //calculation of hours
 dh=sek/600; sek%=600;                          //calculation of ten of minutes
 h=sek/60; sek%=60;                             //calculation of minutes
 ds=sek/10;                                     //calculation of ten of seconds
 s=sek%10;                                      //calculation of seconds
res=((u32)dg<<20)|((u32)g<<16)|((u32)dh<<12)|((u32)h<<8)|((u32)ds<<4)|s;
 return res;
}
int main(){
 a=0x132259;
 b=0x023441;
 while(1) {
  c=ar_op(0,a,b);    // addition
  d=ar_op(1,a,b);    // subtraction
  a++;
  b++;
 }
}
```

The analysis of the program code reveals, that in first case of pogram occupied 67 lines, and in the second - there are 38 lines. The first algorithm is difficult for understanding, second - more accessible.

At the same time the compiled first program has a volume a 906 byte, and the second - 1398 byte.

For comparison the program, which is written with the Ci++ language, in which the string data type is realized, is considered [2]. Unsealing of the program is showed on Listing 2. Results of implementation of arithmetic operations in behalf of algorithm 1.

Listing 2.

```
. include <stdio.h>
void set_time(char* _time, char form, char oper, int value){
    int *tmp_int = new int;
    sscanf(_time, "%d", tmp_int);
    int time = 0;
    time = time + (*tmp int%100) + ((*tmp int%10000)/100)*60 + (*tmp int/10000)*3600;
    delete tmp_int;
    switch(oper){
    case '+':
        switch(form){
        case 'h': time = time + value*3600; break;
        case 'm': time = time + value*60; break;
        case 's': time = time + value; break;
        default: return;
        }
        break;
    case '-':
        switch(form){
        case 'h': time = time - value*3600; break;
        case 'm': time = time - value*60; break;
        case 's': time = time - value; break;
```

```
        default: return;
        }
        break;
    default: return;
    }
    if(time<=0){    sprintf(_time, "000000");    }
    else{
        if((time/3600)>=10){
            sprintf(_time, "%d", (time/3600));
        }
        else{
            sprintf( time, "0%d", (time/3600));
        }
        if(((time%3600)/60)>=10){
            sprintf(_time, "%s%d", _time,  ((time%3600)/60));
        }
        else{
            sprintf(_time, "%s0%d", _time, ((time%3600)/60));
        }
        if((time%60)>=10){
            sprintf(_time, "%s%d", _time, time%60);
        }
        else{
            sprintf( time, "%s0%d",  time, time%60);
        }    }}
int main(){
    char *time = new char[7];
    sprintf(time, "071355");
    set_time(time, 'm', '+', 1);
    set_time(time, 's', '+', 13);
    printf("%s\n", time);
    return 0;
}
```

The operating speed analysis of the offered and considered algorithms proved, that for the algorithm 1 is 7746 operations of addition required, in that time as for the algorithm 2 performance - 1074 operations, the program is represented on Listing 2 requires the yet less number of operations. It is related to the necessity of working of long 32-bit arrays of numeric data at 8-bit AVR microprocessor. At information processing in the time format on 32-bit microprocessors the saving of time is foreseen at using of algorithm 1 in 1,5 time, by comparison to an algorithm 2 and timetable will be commensurable with the use of the Listing 2 program.

Conclusions

Two algorithm and program variants of implementation of addition and deduction operations of numbers, presented in the time format are developed. The program is written with the C language for the AVR family microcontroller.

Performance of algorithm of operations implementation was analysed and researchd by the offered method. An algorithm enables quickly to execute numbers operations.

The first program gives a long code, but occupies a less place in memory. Second - vice versa. At the choice of variant it follows to take into account a program clearness also.

1. *Kurbatova K.A. Microsoft Exel 2003. Concise course / K.A. Kurbatova. – M.: Publishing house «Williams», 2004. – 288 p.*
2. *http://www.cyberforum.ru/cpp-beginners/thread773189.html, Arithmetic operations with information of "time" type.*
3. *http://controllersystems.com/books/praktika_programmirovaniya_atmel_avr/operacii-s-chislami-v-formate-bcd.html, Numbers operations in BCD format.*
4. *Evtifeev A.V. AVR microcontrollers of Classic family byATMEL / A.V. Evtifeev – M.: Publishing house «Dodeka-XXI», 2006. — 288 p.*
5. *Revitch U.V. The practical programming of Atmel AVR microcontrollers in assembly language – SPb.: BHV Perburg, 2011. – 352 p.*
6. *Baranov V.N. Application of AVR microcontrollers: charts, algorithms, programs/ V.N. Baranov.— M.: .: Publishing house «Dodeka-XXI», 2004. –288 p:*
7. *Trampert W. AVR-RISC microcontrollers / W. Trampert..– Translation from German. - K.: "MK-Press", 2006. - 464 p.*