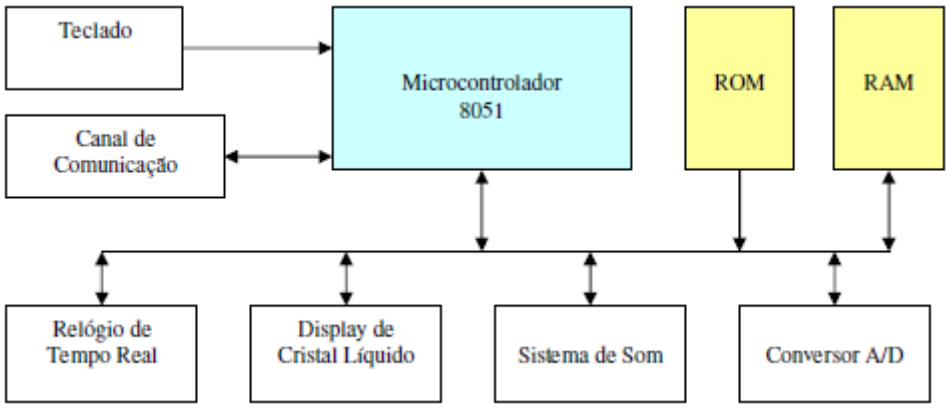
This project contains a program with “C for machines” language that, when compiled, generates assembly code and other types that are recorded in the ROM memory to work without computer input.

With a use of an experimental board with an 8051 microcontroller, showed in this block diagram below:



Translation:

- 8051 microcontroller = Microcontrolador
- Keyboard = Teclado (a 16-key numerical keyboard)
- Communication channel = Canal de Comunicação
- Real-time clock = Relógio de Tempo Real
- LCD Display = Display de Cristal Líquido
- Sound system = Sistema de Som
- A/D Converter = Conversor A/D

The program we made was to make an infusion bomb work, automating it when liquid temperature is adequate (monitored by a LM35 temperature sensor that sends signal via A/D Converter to 8051 microcontroller that deals with the data and turns a heater on or off) and according to programmed schedule (or manually activating it via keyboard). As the program, after compiled and deployed, will be recorded in ROM memory, the board could be embedded with the bomb and work without a computer to adjust clock or alarm, which could be adjusted with this 16-key keyboard (0 – F buttons).

The programming language I used is customized and named “C for machines”, a more familiar language than Assembly, for example. This compilation allows that this program can be “translated” for microcontroller’s language.

Let’s see how the program works, focusing on MAIN function, showed below:

```
void main(void){

    char tecla;
    progrdisplay(); /* programa display */
    line(1);
    cleardisplay();
    IE = 0x81;

    adc_data = 0x00;
    initTempoReal();

    T0 = 1;
    T1 = 1;

    while (1) {
        cleardisplay();
        printDisplay("Escolha:");
        tecla = NovaTeclaKeyDown();
          if (tecla == '0') prog();
          if (tecla == '1') progAlarm();
          if (tecla == '2') liga_motor();
          if (tecla == '3') desliga_motor();
          if (tecla == '4') liga_aquecedor();
          if (tecla == '5') desliga_aquecedor();
          if (tecla == '7') mostraDisplay();
    };
}
```

With the keyboard properly programmed to receive an input from a determined key and the LCD display working as an output display, mapped in the ROM memory, the program enter in idle state, where it waits for a pressing button, just like the computer waits for a mouse movement or pressing button from a keyboard to perform new activity.

IE is the register that initiates the real-time clock, while the function initTempoReal() configures how the clock works (12/24 hours display, bytes or integer data type). The adc_data variable receives the data from A/D converter (i.e. from temperature sensor), where a value attribution to this variable triggers a new temperature sensor reading, while the lines progrdisplay(), line(1) and cleardisplay() are for the display. T0 and T1 variables are output variables for peripherals (heater and infusion bomb's motor), which can be manually activated by the user (with "2" key and "5" key that allows these activations) or automatically via alarm.

The "7" key shows time and liquid temperature on display and "0" key allows the user to change clock's time, while "1" key allows the user to change alarm's time. The function for clock's time change is showed in the appendix A – prog.c.

To change a time, the user can change a date (year, month, day), weekday and the actual time (hour, minute, second). "0-9" keys are used for the numbers' input, while "E" key confirms the change and "F" key erase the last digit inputted. regB is the register variable that verifies if the real-time clock can works or not.

Note that to change the time, the variable receives values to stop the clock and, after the adjustment, the clock works again with the new values.

Appendix A – prog.c

```
void prog()
{
   char auxIE, aux[2], tecla;
         auxIE = IE;
         IE &= 0xEF;
         regB |= 0x80;

   INICIO_PROG_ANO:

         cleardisplay();
         printDisplay("Ano(00-99): ");
         aux[0] = Key_ASCII2BIN(NovaTeclaKeyDown());
         while ( (aux[0] < 0 || aux[0] > 9) && aux[0] != 0xF )
      aux[0] = Key_ASCII2BIN(NovaTeclaKeyDown());
   if (aux[0] == 0xF)
      goto INICIO_PROG_ANO;

         displaydata(aux[0]+0x30);
         aux[1] = Key_ASCII2BIN(NovaTeclaKeyDown());
         while ( (aux[1] < 0 || aux[1] > 9) && aux[1] != 0xF)
      aux[1] = Key_ASCII2BIN(NovaTeclaKeyDown());
   if (aux[1] == 0xF)
      goto INICIO_PROG_ANO;

         displaydata(aux[1]+0x30);

         tecla = Key_ASCII2BIN(NovaTeclaKeyDown());
         while (tecla != 0xE && tecla != 0xF)
      tecla = Key_ASCII2BIN(NovaTeclaKeyDown());
         if (tecla == 0xF)
      goto INICIO_PROG_ANO;
         if (tecla == 0xE)
      year = aux[0]*16 + aux[1];

   INICIO_PROG_MES:

         cleardisplay();
         printDisplay("Mes(00-12): ");
         aux[0] = Key_ASCII2BIN(NovaTeclaKeyDown());
         while ( (aux[0] < 0 || aux[0] > 9) && aux[0] != 0xF )
      aux[0] = Key_ASCII2BIN(NovaTeclaKeyDown());
   if (aux[0] == 0xF)
      goto INICIO_PROG_MES;
```

```
        displaydata(aux[0] + 0x30);
        aux[1] = Key_ASCII2BIN(NovaTeclaKeyDown());
        while ( (aux[1] < 0 || aux[1] > 9) && aux[1] != 0xF )
    aux[1] = Key_ASCII2BIN(NovaTeclaKeyDown());
if (aux[1] == 0xF)
    goto INICIO_PROG_MES;

        displaydata(aux[1] + 0x30);
        tecla = Key_ASCII2BIN(NovaTeclaKeyDown());
        while (tecla != 0xE && tecla != 0xF)
    tecla = Key_ASCII2BIN(NovaTeclaKeyDown());
        if (tecla == 0xF)
    goto INICIO_PROG_MES;
        if (tecla == 0xE)
    mounth = aux[0]*16 + aux[1];

INICIO_PROG_DIA:

        cleardisplay();
        printDisplay("Dia(00-31): ");
        aux[0] = Key_ASCII2BIN(NovaTeclaKeyDown());
        while ( (aux[0] < 0 || aux[0] > 9) && aux[0] != 0xF )
    aux[0] = Key_ASCII2BIN(NovaTeclaKeyDown());
if (aux[0] == 0xF)
    goto INICIO_PROG_DIA;

        displaydata(aux[0] + 0x30);
        aux[1] = Key_ASCII2BIN(NovaTeclaKeyDown());
        while ( (aux[1] < 0 || aux[1] > 9) && aux[1] != 0xF )
    aux[1] = Key_ASCII2BIN(NovaTeclaKeyDown());
if (aux[1] == 0xF)
    goto INICIO_PROG_DIA;

        displaydata(aux[1] + 0x30);
        tecla = Key_ASCII2BIN(NovaTeclaKeyDown());
        while (tecla != 0xE && tecla != 0xF)
    tecla = Key_ASCII2BIN(NovaTeclaKeyDown());
        if (tecla == 0xF)
    goto INICIO_PROG_DIA;
        if (tecla == 0xE)
    dateOfMounth = aux[0]*16 + aux[1];

INICIO_PROG_SEMANA:
```

```
        cleardisplay();
        printDisplay("DiaSem(01-07):");
        aux[0] = Key_ASCII2BIN(NovaTeclaKeyDown());
        while ( (aux[0] < 0 || aux[0] > 9) && aux[0] != 0xF )
    aux[0] = Key_ASCII2BIN(NovaTeclaKeyDown());
if (aux[0] == 0xF)
    goto INICIO_PROG_SEMANA;

        displaydata(aux[0] + 0x30);
        aux[1] = Key_ASCII2BIN(NovaTeclaKeyDown());
        while ( (aux[1] < 0 || aux[1] > 9) && aux[1] != 0xF )
    aux[1] = Key_ASCII2BIN(NovaTeclaKeyDown());
if (aux[1] == 0xF)
    goto INICIO_PROG_SEMANA;

        displaydata(aux[1] + 0x30);
        tecla = Key_ASCII2BIN(NovaTeclaKeyDown());
        while (tecla != 0xE && tecla != 0xF)
    tecla = Key_ASCII2BIN(NovaTeclaKeyDown());
        if (tecla == 0xF)
    goto INICIO_PROG_SEMANA;
        if (tecla == 0xE)
    dayOfWeek = aux[0]*16 + aux[1];

INICIO_PROG_HORA:

        cleardisplay();
        printDisplay("Hora(00-23): ");
        aux[0] = Key_ASCII2BIN(NovaTeclaKeyDown());
        while ( (aux[0] < 0 || aux[0] > 9) && aux[0] != 0xF )
    aux[0] = Key_ASCII2BIN(NovaTeclaKeyDown());
if (aux[0] == 0xF)
    goto INICIO_PROG_HORA;

        displaydata(aux[0] + 0x30);
        aux[1] = Key_ASCII2BIN(NovaTeclaKeyDown());
        while ( (aux[1] < 0 || aux[1] > 9) && aux[1] != 0xF )
    aux[1] = Key_ASCII2BIN(NovaTeclaKeyDown());
if (aux[1] == 0xF)
    goto INICIO_PROG_HORA;

        displaydata(aux[1] + 0x30);
        tecla = Key_ASCII2BIN(NovaTeclaKeyDown());
        while (tecla != 0xE && tecla != 0xF)
    tecla = Key_ASCII2BIN(NovaTeclaKeyDown());
```

```
        if (tecla == 0xF)
    goto INICIO_PROG_HORA;
        if (tecla == 0xE)
    hours = aux[0]*16 + aux[1];

INICIO_PROG_MIN:

        cleardisplay();
        printDisplay("Min(00-59): ");
        aux[0] = Key_ASCII2BIN(NovaTeclaKeyDown());
        while ( (aux[0] < 0 || aux[0] > 9) && aux[0] != 0xF )
    aux[0] = Key_ASCII2BIN(NovaTeclaKeyDown());
if (aux[0] == 0xF)
    goto INICIO_PROG_MIN;

        displaydata(aux[0] + 0x30);
        aux[1] = Key_ASCII2BIN(NovaTeclaKeyDown());
        while ( (aux[1] < 0 || aux[1] > 9) && aux[1] != 0xF )
    aux[1] = Key_ASCII2BIN(NovaTeclaKeyDown());
if (aux[1] == 0xF)
    goto INICIO_PROG_MIN;

        displaydata(aux[1] + 0x30);
        tecla = Key_ASCII2BIN(NovaTeclaKeyDown());
        while (tecla != 0xE && tecla != 0xF)
    tecla = Key_ASCII2BIN(NovaTeclaKeyDown());
        if (tecla == 0xF)
    goto INICIO_PROG_MIN;
        if (tecla == 0xE)
    minutes = aux[0]*16 + aux[1];

INICIO_PROG_SEG:

        cleardisplay();
        printDisplay("Seg(00-59): ");
        aux[0] = Key_ASCII2BIN(NovaTeclaKeyDown());
        while ( (aux[0] < 0 || aux[0] > 9) && aux[0] != 0xF )
    aux[0] = Key_ASCII2BIN(NovaTeclaKeyDown());
if (aux[0] == 0xF)
    goto INICIO_PROG_SEG;

        displaydata(aux[0] + 0x30);
        aux[1] = Key_ASCII2BIN(NovaTeclaKeyDown());
        while ( (aux[1] < 0 || aux[1] > 9) && aux[1] != 0xF )
    aux[1] = Key_ASCII2BIN(NovaTeclaKeyDown());
```

```
  if (aux[1] == 0xF)
    goto INICIO_PROG_SEG;

        displaydata(aux[1] + 0x30);
        tecla = Key_ASCII2BIN(NovaTeclaKeyDown());
        while (tecla != 0xE && tecla != 0xF)
    tecla = Key_ASCII2BIN(NovaTeclaKeyDown());
        if (tecla == 0xF)
    goto INICIO_PROG_SEG;
        if (tecla == 0xE)
    seconds = aux[0]*16 + aux[1];


        IE = auxIE;
        regB &= 0x7F;
}
```