Final Project Report
Embedded Software & Design

Digital Alarm Clock using Atmega128 Microcontroller.

Submitted by:
Team: WaduDev 16-3
U1610146 Mirzashomol Karshiev
U1610131 Madiyor Abdukhashimov
U1610137 Mardon Zarifjonov

Inha University in Tashkent
2020

Abstract:

We have been given the task to make Digital Alarm Clock using Amtega128 microcontroller. We were provided with SimulIDE software simulation tool that is used for simulating the circuitry (kit) and ATMEGA128.simu file to imitate real kit. Our digital clock has three modes: Normal Time, Alarm and Stop watch.

We have completed the task by the help of knowledge obtained by professor. The themes that were necessary to complete the task: LCD Keyboard and Interfacing, Interrupt and Timer.
The necessary software tool for writing code to the kit is Atmel Studio. PORTD was used for switch PD0(INT0) ,PD2(INT2) and PD3(INT3). PORTB was used for LED blinking for Alarm. PORTA was used for Character LCD.

Our digital clock starts with setting the time using switches PD2 and PD3. Using PD0 switch it can change the mode. Stop watch is started right after switching by using PD0 and stopped by PD3 switch. Alarm is set with PD2 and PD3 switch for go.

Acknowledgements:

The goal of this project was closely understand the working principle behind embedded devices by learning AVR C Programming on Atmel Studio. We would like thank our professor for detailed explanation of necessary themes that was important in completing our project. And we would like thank our team members for their dedication and hard work.

Table of contents:

1. Introduction:

As we all know this era is era of ICT. For that reason, everything is being digitalized and the embedded devices are playing important role in convergence of Intelligent services (IoT). It shows the necessity of knowledge.
The main goal of this report is to clearly explain to readers how this Digital Alarm Clock is created using Atmega128 Microcontroller and what kind of difficulties we had during the project. This report will tell you first what algorithm behind this Digital Alarm Clock and it shows Instruction Manual. At the end it will tell how to divide the task for efficient outcome.

2. Algorithm:

We wrote algorithm for Digital Alarm Clock based on knowledge of AVR C Programming that we obtained from our professor and from the book *"the avr microcontroller and embedded system using assembly and c" by Muhammad Ali, Sarmad Naimi and Sepher Naimi.*

Firstly, we had to use several libraries for our project they are:

```
11  #include <avr/io.h>
12  #include <avr/interrupt.h>
13  #include "_main.h"
14  #include "lcd_n.h"
```

avr/io for input/output programming interrupts are used for pressing sw.

lcd display functions.

ISR Interrupt :

```
271 //<----------Timer0/Counter----------->
272 ISR(TIMER0_COMP_vect)
273 {
274     
275     cnt++;
276     if(cnt==144)
277     {
278         cnt=0;
279         sec++;
280         s_sec++;
281         if(sec>=60)
282         {
283             min++;
284             s_min++;
285             sec=0;
286             s_sec=0;
287         }
288         if(min>=60)
289         {
290             hour++;
291             s_hour++;
292             min=0;
293             s_min=0;
294         }
295         if (hour>=24)
296         {
297             hour=0;
298             s_hour=0;
299         }
300     }
301 }
```

ISR() is timer counter. And it is interrupted when compared to OCR0. Accordingly, time is set.

The main function:

```
529 int main(void)
530 {
531     Init();                 //Initializes all Interrupts, Ports, LCD and variables.
532     WelcomeDisplay();       //Welcoming Message
533     Set_Time();             //Sets the Timer initially.
534
535     while (1)
536     {
537         switch(bool)        //Based on Switch (PD0) it will switch between modes
538         {
539             case 0:
540                 Time();     //Normal Time mode
541                 break;
542             case 1:
543                 Stop_Watch();//Stop watch mode
544                 break;
545             case 2:
546                 Alarm();    //Alarm mode
547                 break;
548             default:
549                 break;
550         }
551
552     }
553 }
```

Initially we initialize all our variables, interrupts, ports and lcd display by function Init(); which contains all code. Then WelcomeDisplay() method shows on LCD welcoming message “WELCOME TO WaduDev 16-3”. After that we have to set the timer using Set_Time() function. After setting the timer it automatically goes inside loop and goes to Time() function because bool initially is 0. The bool is changed according to PD0 or INT0. Whenever, PD0 is pressed bool is incremented. It is (PD0) used as selection between modes.

The Init function:

```
436 void Init(void)
437 {
438     //Normal Time Initialization
439     cnt=0;
440     sec=min=0;
441     hour=12;
442     bool=0;
443
444     //StopWatch Time Initialization
445     s_sec=0;
446     s_min=0;
447     s_hour=0;
448     ch=0;
449
450     //Initialize Interrupt
451     cli();              //Clear Interrupts
452     EIMSK = 0x0f;       //ENABLE INT0 for SWITCH
453     EICRA = 0xAA;       //INT0-->falling edge
454     sei();              //Set Interrupt bit
455
456     //Initialize LCD
457     PortInit();
458     DDRD = 0x00;    //PORTD is input (SWITCH)
459     DDRB = 0xff;    //LED output
460     PORTB=0xff;
461     LCD_Init();     //Initialize LCD by LCD_Comm
462     LCD_Clear();    //Clear the Display
463     LCD_pos(0,0);   //Move position to 0,0
464     LCD_STR(str);   //OUTPUT: TIME FUNCTION
465
466     //Initialize Timer
467     Init_timer0();}
```

The Init function initializes variables of Normal Time, Stop watch.

It first clears interrupts by cli(); then by EIMSK it sets INT0, INT1, INT2 and INT3.

It initializes Ports of LCD and sets PORTD as input, PORTB as output for LED.

Init_timer0() is called last that initializes the timer.

```
61 //<----------Initialize Timer0----------->
62 void Init_timer0(void)
63 {
64     OCR0=99;            //count 0 to 99
65     TIMSK=0x02;         //time0/counter compare interrupt enable
66 }
67
```

OCR0 is set for CTC Timer. TIMSK for compare interrupt.

The Welcome Display function:

```
508  //<----------Welcoming Page----------->
509 void WelcomeDisplay()
510 {
511     LCD_Clear();
512     for(char i=0;i<3;i++)
513     {
514         LCD_pos(3,0);
515         LCD_STR("WELCOME TO");
516         LCD_pos(2,1);
517         LCD_STR("WaduDeV 16-3");
518         _delay_ms(500);
519
520         LCD_Clear();
521         _delay_ms(500);
522
523     }
524 }
```

The WelcomeDisplay() function it shows three times the welcoming message “Welcome to WaduDev 16-3” by the help of LCD_pos() and LCD_STR() functions that are used for position of string.

The Set time function:

```
//<----------Setting Normal Time----------->
void Set_Time()
{   LCD_Clear();
    while(1)
    {
        if (t_bool ==1)
        {
            LCD_Clear();
            _delay_ms(300);
            break;
        }
        else{
        LCD_pos(0,0);
        LCD_STR("SETTING TIME:");

        LCD_pos(0,1);
        LCD_CHAR((t_hour)/10+'0');
        LCD_CHAR((t_hour)%10+'0');
        LCD_CHAR(':');

        LCD_pos(3,1);          //shows min 00:
        LCD_CHAR((t_min/10)+'0');
        LCD_CHAR((t_min%10)+'0');
        LCD_CHAR(':');

        LCD_pos(6,1);          //shows sec 00
        LCD_CHAR((t_sec/10)+'0');
        LCD_CHAR((t_sec%10)+'0');

        LCD_pos(10,1);
        LCD_STR(days);
}}}
```

Set_Time() function is used initially to set the Normal Time. First it clears LCD display then it sets the Timer using PD2(INT2). Whenever PD2 is pressed it changes the hour, min, sec and day of the week by SIGNAL(INT2_vect) interrupt that has logic to do so.

```
//<----------PD2(INT2) For Setting Timer----------->
SIGNAL(INT2_vect)
{
    //Setting the Time
    if(set_bool==0)
    {
        t_hour++;
        if (t_hour==24)
            t_hour=0;
    }
     if(set_bool==1)
    {
        t_min++;
        if (t_min==60)
            t_min=0;
    }
     if(set_bool==2)
    {
        t_sec++;
        if (t_sec==60)
            t_sec=0;
    }
    if(set_bool==3)
    {
        switch(day_chooser)
        {
            case 0:
                days[0]='M'; days[1]='o';days[2]='n';       //Monday
                break;
            case 1:
                days[0]='T';days[1]='u';days[2]='e';        //Tuesday
                break;
```

SIGNAL(INT2_vect) is used for PD2 switch. When it is pressed it will increment hour, min, sec and day of the week to set the Normal Time (Initial Time).

```
            case 2:
                days[0]='W';days[1]='e';days[2]='d';        //Wednesday
                break;
            case 3:
                days[0]='T';days[1]='h';days[2]='u';        //Thursday
                break;
            case 4:
                days[0]='F';days[1]='r';days[2]='i';        //Friday
                break;
            case 5:
                days[0]='S';days[1]='a';days[2]='t';        //Saturday
                break;
            case 6:
                days[0]='S';days[1]='u';days[2]='n';        //Sunday
                break;
            case 7:
                days[0]='M';days[1]='o';days[2]='n';        //Monday
                day_chooser=0;
                break;
        }
        day_chooser++;
    }
```

The Stop watch function:

```
 //<----------Stopwatch Function----------->
void Stop_Watch(void)
{
        if (stop_watch_bool==1)
        {
            LCD_pos(0,0);
            LCD_STR(OFF);
        }else{
            LCD_pos(0,0);
            LCD_STR(ON);
        }

        LCD_pos(0,1);
        LCD_CHAR((s_hour)/10+'0');
        LCD_CHAR((s_hour)%10+'0');
        LCD_CHAR(':');

        LCD_pos(3,1);         //shows min 00:
        LCD_CHAR((s_min/10)+'0');
        LCD_CHAR((s_min%10)+'0');
        LCD_CHAR(':');

        LCD_pos(6,1);         //shows sec 00
        LCD_CHAR((s_sec/10)+'0');
        LCD_CHAR((s_sec%10)+'0');

        LCD_CHAR('.');
        LCD_pos(9,1);
        LCD_CHAR(((cnt/10)%10)+'0');
        LCD_CHAR((cnt%10)+'0');
}
```

The Stop_Watch() function is started right after pressing PD0(INT0) switch, SIGNAL(INT0_vect).

Stop watch shows the hour, min, sec starting from 0 sec.

Stop watch is stopped when PD3(INT3) is pressed.W

```
69  //<----------PD1(INT0) For Switching Between Modes----------->
70  SIGNAL(INT0_vect)
71  {
72      TCCR0=0x0f;
73      bool++;
74      stop_watch_bool=0;
75
76
77      if (bool==3)
78      {
79          t_bool=1;
80          bool=0;
81      }
82
83      if (bool==2)
84      {
85          sw_counter++;
86          t_bool=0;
87          alarm_bool=0;
88          day_chooser=0;
89          t_hour=t_min=t_sec=0;
90      }
91
92      if (bool==1)
93         {
94             s_hour=s_min=s_sec=0;
95             stp_bool=1;
96         }
97
98  }
```

Whenever PD0 is pressed INT0 is handled then bool is incremented which needed to switch between modes.

When bool is equal to 1. It stops the stop watch with help of Boolean stp_bool

When bool is equal 2, the mode is changed to alarm.

The Alarm function:

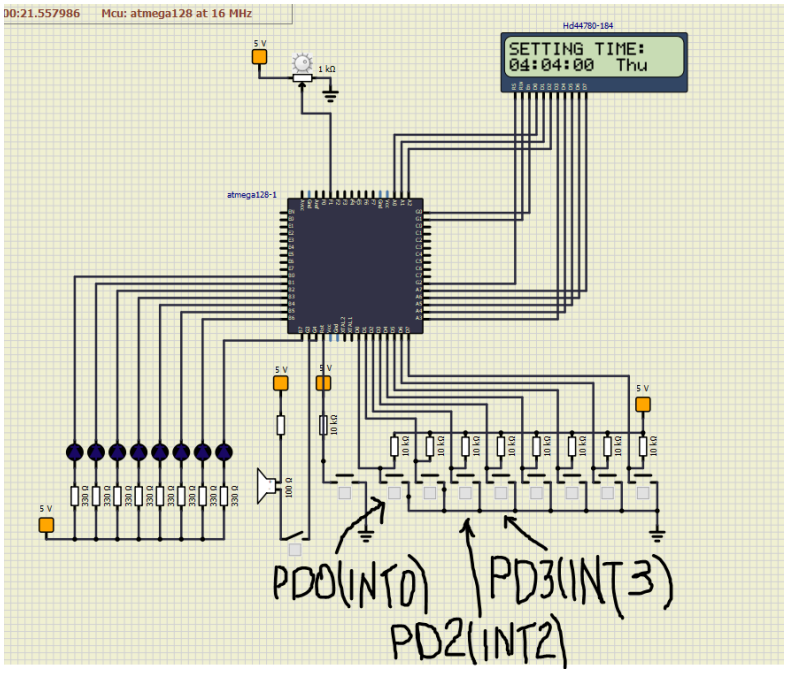
```
394  //<----------Alarm Function----------->
395  void Alarm(void)
396  {
397      LCD_Clear();
398
399      while(1)
400      {
401          if (t_bool==1)
402          {
403              LCD_Clear();
404              break;
405          }
406
407          else{
408              LCD_pos(0,0);
409              LCD_STR("SETTING ALARM:");
410
411              LCD_pos(0,1);
412              LCD_CHAR((t_hour)/10+'0');
413              LCD_CHAR((t_hour)%10+'0');
414              LCD_CHAR(':');
415
416              LCD_pos(3,1);          //shows min 00:
417              LCD_CHAR((t_min/10)+'0');
418              LCD_CHAR((t_min%10)+'0');
419              LCD_CHAR(':');
420
421              LCD_pos(6,1);          //shows sec 00
422              LCD_CHAR((t_sec/10)+'0');
423              LCD_CHAR((t_sec%10)+'0');
424
425              LCD_pos(10,1);
426              LCD_STR(days);}
427      }}
428
```

The Alarm function is the same as Set_Time function. We set Alarm using PD2(INT2 and PD3(INT3). INT2 for incrementing and INT3 for setting.
After setting the alarm the alarm time stored in variables.
It is compared inside Time() function, whenever time is equal to alarm it will blink LEDs.

Finally PD3 for setting time, alarm and stop watch :

```
211  //<----------PD3(INT3) For Stopping Timer----------->
212 SIGNAL(INT3_vect)
213 {
214     if (stp_bool==1 && bool==1)
215     {
216         LCD_Clear();
217         stop_watch_bool=1;
218         TCCR0=0x00;             //stop the timer
219     }
220
221     //for setting the time
222     if (set_bool==0)
223     {
224         hour=t_hour;
225         set_bool=1;
226     }
227     else if (set_bool==1)
228     {
229         min=t_min;
230         set_bool++;
231     }
232     else if (set_bool==2)
233     {
234
235         sec=t_sec;
236         set_bool++;
237     }else if (set_bool==3)
238     {
239         set_bool++;
240         TCCR0=0x0f;
241         t_bool=1;
242     }
243
244
245     //For setting the alarm
246     if (alarm_bool==0)
247     {
248         a_hour=t_hour;
249         alarm_bool++;
250     }
251     else if (alarm_bool==1)
252     {
253         a_min=t_min;
254         alarm_bool++;
255     }
256     else if (alarm_bool==2)
257     {
258         a_sec=t_sec;
259         alarm_bool++;
260     }else if (alarm_bool==3)
261     {
262         alarm_bool++;
263
264         TCCR0=0x0f;
265         t_bool=1;
266         bool=0;
267     }
268
269 }
```

3. User Manual:



START PROGRAM:

1. Set Time with PD2 (increment) and PD3 (set).
2. PD0 can switch between modes.
3. When you are in Stop watch to stop it use PD3.
4. When you are in Alarm, for setting use PD2(increment) and PD3 (set).

4. Division of tasks:

We had to divide our tasks into 3 parts so that our job would be efficient.

Task Division		
Team Name	WaduDev 16-3	**Date:** May 1, 2020
Role	**Name**	**Main Responsibility**
Main Coder	**Mirzashomol Karshiev**	Implemented the C code on Atmel studio.
Facilitator	**Mardon Zarifjonov**	Helped with logic of program and report, requirement of analysis.
Coder	**Madiyor Abdukhashimov**	Helped with optimizing the code and fixing bugs, SW design specifications.