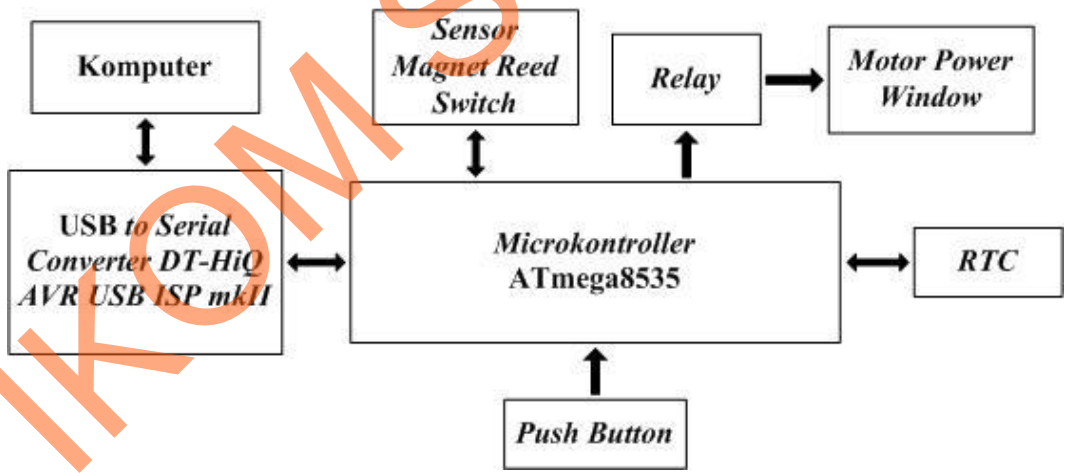
# BAB IV

# PEMBAHASAN

## 4.1 Perangkat Keras

Informasi waktu yang akan ditunjukkan oleh jarum dan motor power window yang telah dimodifikasi menggunakan gear akan digunakan sebagai penggerak jarum jam. Informasi mengenai waktu aktual akan diambil dari RTC yang digunakan sebagai pewaktu pada sistem. Parameter-parameter yang diperlukan oleh sistem untuk bekerja dengan benar akan diberikan dengan bantuan computer melalui komunikasi serial. Blok diagram keseluruhan sistem terdapat pada gambar 4.1.




Gambar 4.1 Blok Diagram Keseluruhan Sistem Jam

*Microcontroller* yang digunakan pada sistem ini, yaitu ATmega8535 dan berfungsi sebagai pengontrol sistem. Pengontrol yang dilakukan meliputi pembacaan waktu serta penulisan register-register pada RTC, dalam sistem ini

tipe RTC yang digunakan adalah DS1307, menerima sinyal input dari sensor, memberikan *trigger* pada relay yang akan memicu motor power window, mendeteksi penekanan tombol dari *user*, dan menerima serta mengirimkan data serial ke komputer yang digunakan untuk mengatur parameter-parameter yang dibutuhkan.

RTC digunakan sebagai acuan waktu pada sistem. Informasi yang disediakan adalah detik, menit, jam, tanggal, bulan, dan tahun. Pengaksesan RTC oleh *microcontroller* dilakukan secara serial dengan protokol komunikasi $I^2C$. Sensor *magnet reed switch* berfungsi sebagai pendeteksi lokasi jarum jam. Sensor *magnet reed switch* akan memberikan input tegangan setiap 15 menit pada *microcontroller*.

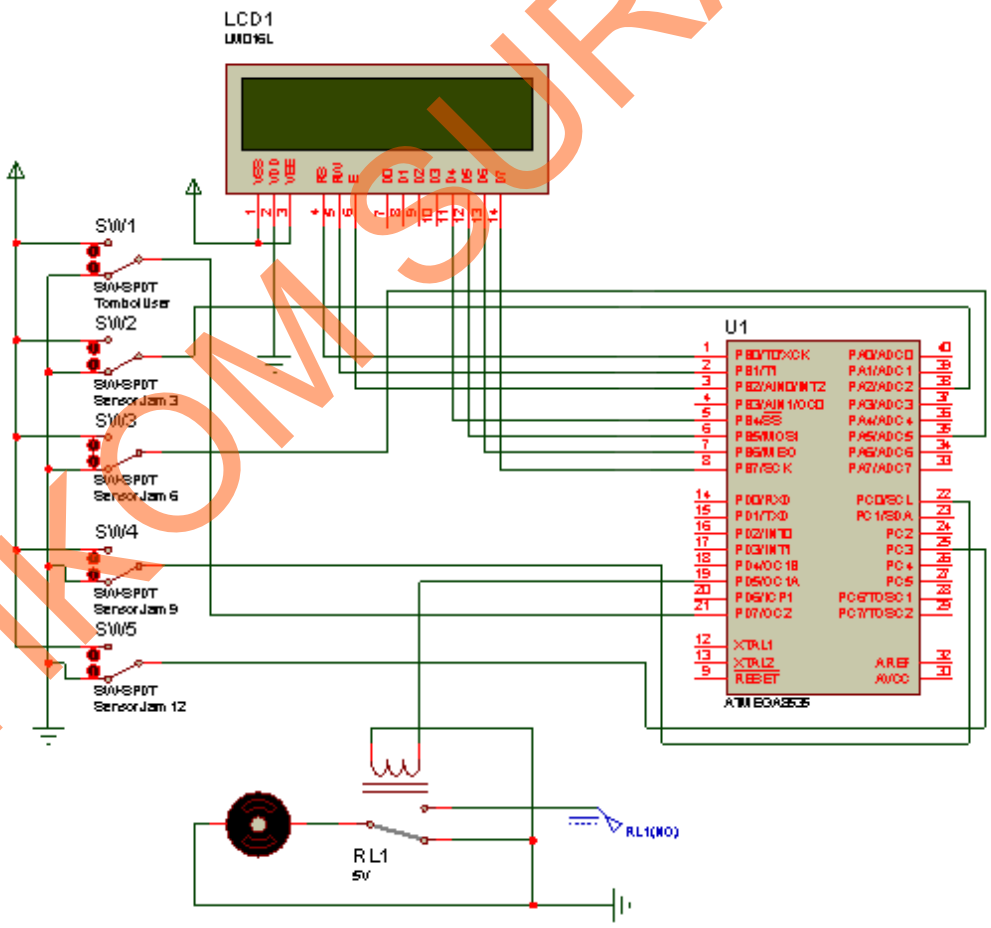
Agar sistem ini dapat bekerja, dibutuhkan pengaturan beberapa parameter. Pengaturan parameter menggunakan komputer yang akan dihubungkan dengan *microcontroller* melalui USB *to Serial Converter DT-HiQ AVR USB ISP mkII*. Komunikasi antara komputer dengan *microcontroller* dilakukan secara serial.

### 4.1.1 *Minimum System* ATmega8535

Pada rangkaian *minimum system* ATmega8535 yang digunakan pada sistem dapat diilustrasikan seperti gambar 4.2. Port A yaitu PA2 digunakan sebagai masukkan untuk sensor jam yang menunjukkan pada angka 3 (menunjukkan waktu 15 menit pertama) dan PA5 digunakan sebagai masukkan untuk sensor jam yang menunjukkan pada angka 6 (menunjukkan waktu 15 menit kedua). Sama halnya pada Port A, pada Port C yaitu PC0 digunakan sebagai masukkan untuk sensor jam yang

menunjukkan pada angka 9 (menunjukkan waktu 15 menit ketiga) dan PC3 digunakan sebagai masukkan untuk sensor jam yang menunjukkan pada angka 12 (menunjukkan waktu 15 menit keempat).

Pin PB0-PB2 akan dihubungkan dengan Pin 4-5 pada LCD yang digunakan sebagai *Register Select*, *Read/Write*, dan *Enable*. Pin PB5-PB8 akan dihubungkan dengan Pin 11-14 pada LCD yang digunakan sebagai jalur data untuk mengeluarkan outputan. Pin PD7 digunakan untuk mendeteksi penekanan tombol dari user. Pin PD5 akan dihubungkan dengan *relay* dimana *relay* akan memicu motor untuk bergerak disetiap menitnya.

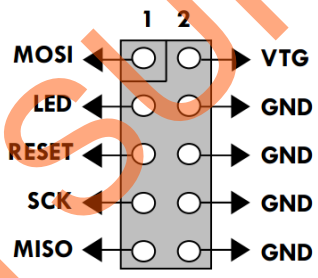



Gambar 4.2 Skema Rangkaian *Minimum System* ATmega8535

Komunikasi dengan RTC dilakukan dengan protokol $I^2C$ melalui pin PD2 sebagai jalur *clock* untuk sinkronisasi komunikasi dan pin PD3 sebagai jalur pertukaran data. Pin VCC diberi masukkan tegangan operasi berkisar antara 4.5 V sampai dengan 5.5 V.

### 4.1.2 Downloader

Untuk melakukan *download* program digunakan perangkat bantu AVR USB ISP yang akan dihubungkan dengan *port* USB (*Universal Serial Bus*) pada computer. Sebelum *downloader* dapat digunakan perlu dikakukan instalasi *drver* terlebih dahulu. Konfigurasi *pinout* dan keterangan dari *downloader* terdapat pada Gambar 4.3 dan Tabel 4.1.



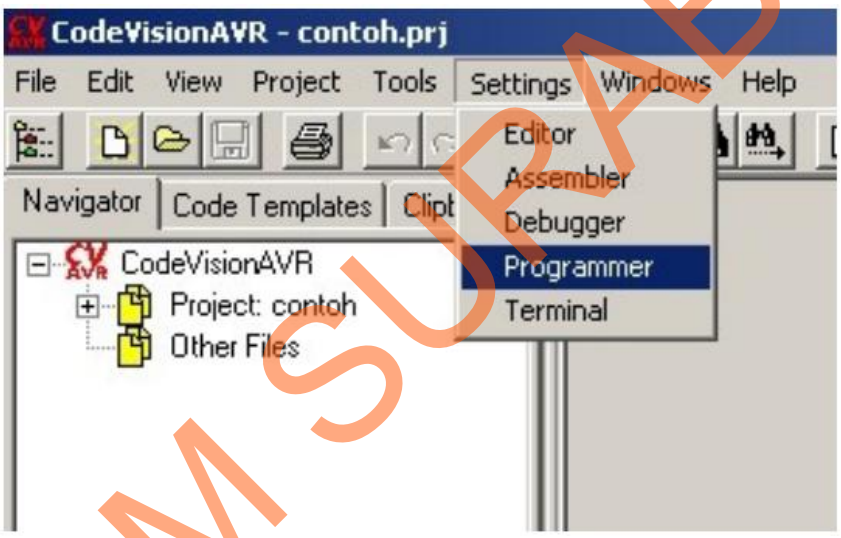

Gambar 4.3 Pinout AVR USB ISP
(INNOVATIVE ELECTRONICS, 2009)

Tabel 4.1 Keterangan *pinout* AVR USB ISP

| Nama | No. Pin | I/O | Keterangan |
|---|---|---|---|
| VTG | 2 | - | Catu daya dari target *board* (2.7-5.5 V) |
| GND | 4, 6, 8, 10 | - | Titik referensi |
| LED | 3 | Output | Sinyal kontrol untuk LED (*Lighting Emitting Diode*) atau *multiplexer* (*optional*) |
| MOSI | 1 | Output | *Command* dan data dari AVR USB ISP ke target AVR |
| MISO | 9 | Input | Data dari target AVR ke AVR USB ISP |
| SCK | 7 | Output | Serial *Clock*, dikendalikan oleh AVR USB ISP |
| RESET | 5 | Output | *Reset*¸dikendalikan oleh AVR USB ISP |

Sumber: INNOVATIVE ELECTRONICS (2009)

Pin MOSI, pin MISO, pin SCK, pin RESET dan pin VTG pada AVR USB ISP masing-masing akan dihubungkan pada pin MOSI, pin MISO, pin SCK, pin RESET dan pin VCC yang terpadat pada *microcontroller*. Program editor dan *compiler* yang digunakan untuk pembuatan program adalah *Code Vision AVR*. Pengaturan penggunaan *downloader* pada *Code Vision AVR* dilakukan dengan memmilih menu *Setting*, kemudian pilihan *Programmer* seperti yang ditunjukkan pada Gambar 4.4



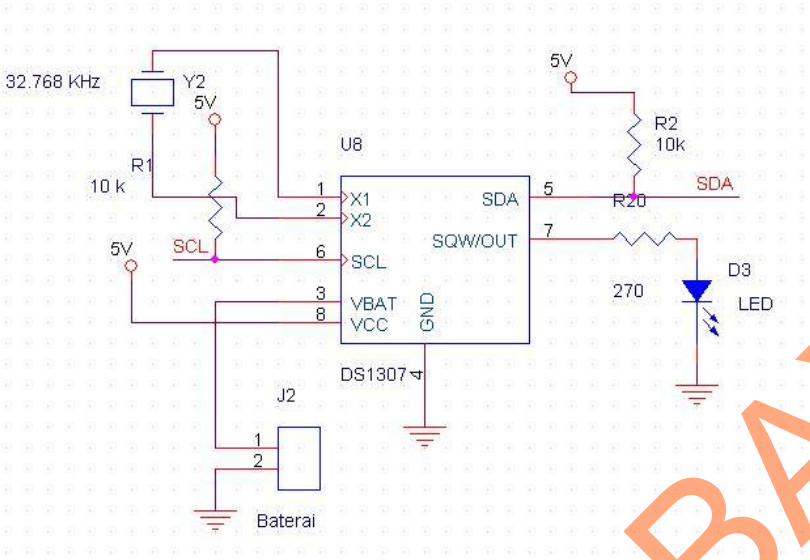

Gambar 4.4 Pemilihan *Programmer* pada menu *Setting* di *Code Vision AVR*

Setelah memilih *Programmer* pada menu *Setting*, akan muncul *window Programmer Setting* seperti pada Gambar 4.5, yang dilanjutkan dengan memilih tipe *programmer AVR* yaitu Atmel AVRISP MkII (USB).


Gambar 4.5 *Window Programmer Setting* pada *Code Vision AVR*

### 4.1.3 RTC DS1307

Rangkaian RTC DS1307 terdapat pada Gambar 4.6. Membutuhkan nilai tegangan dari catu daya utama berkisar 4.5 V - 5.5 V, sedangkan catu daya cadangan (baterai) memiliki nilai tegangan 3V. Jalur komunikasi RTC dilakukan dengan protokol $I^2C$ melalui pin PD2 (yang terhubung dengan pin SDA pada RTC) (yang terhubung dengan pin SCL pada RTC) Pin SDA dan SCL yang berada pada RTC masing-masing diberi resistor *pull-up*. RTC membutuhkan rangkaian osilator eksternal yang hanya terdiri dari *crystal* dengan nilai frekuensi 32.768 kHz yang dihubungkan pada pin 1 dan pin 2 tanpa tambahan kapasitor.




Gambar 4.6 Rangkaian RTC DS1307

Penggunaan RTC harus terlebih dahulu diawali dengan inisialisasi *register control*. Pengaturan *register control* RTC pada program dapat dilakukan dengan menggunakan fungsi sebagai berikut:

```
// DS1307 Real Time Clock initialization
// Square wave output on pin SQW/OUT: On
// Square wave frequency: 1Hz
rtc_init(0,1,0);
```

Pengaturan data waktu pada program dapat dilakukan dengan menggunakan fungsi-fungsi sebagai berikut:

```
rtc_set_time (15,8,20);
```

Pembacaan data waktu pada program dapat dilakukan dengan menggunakan fungsi-fungsi sebagai berikut:

```
rtc_get_time(&jam, &menit, &detik);
```

## 4.2 Perangkat Lunak

Perangkat lunak pada sistem ini hanya dibuat untuk *microcontroller*. Dan untuk menguji apakah program berhasil atau tidak akan dijalankan terlebih dahulu pada simulasi yang telah dibuat pada *Proteus 7 Profesional* sebelum dijalankan pada perangkat yang sebenarnya.

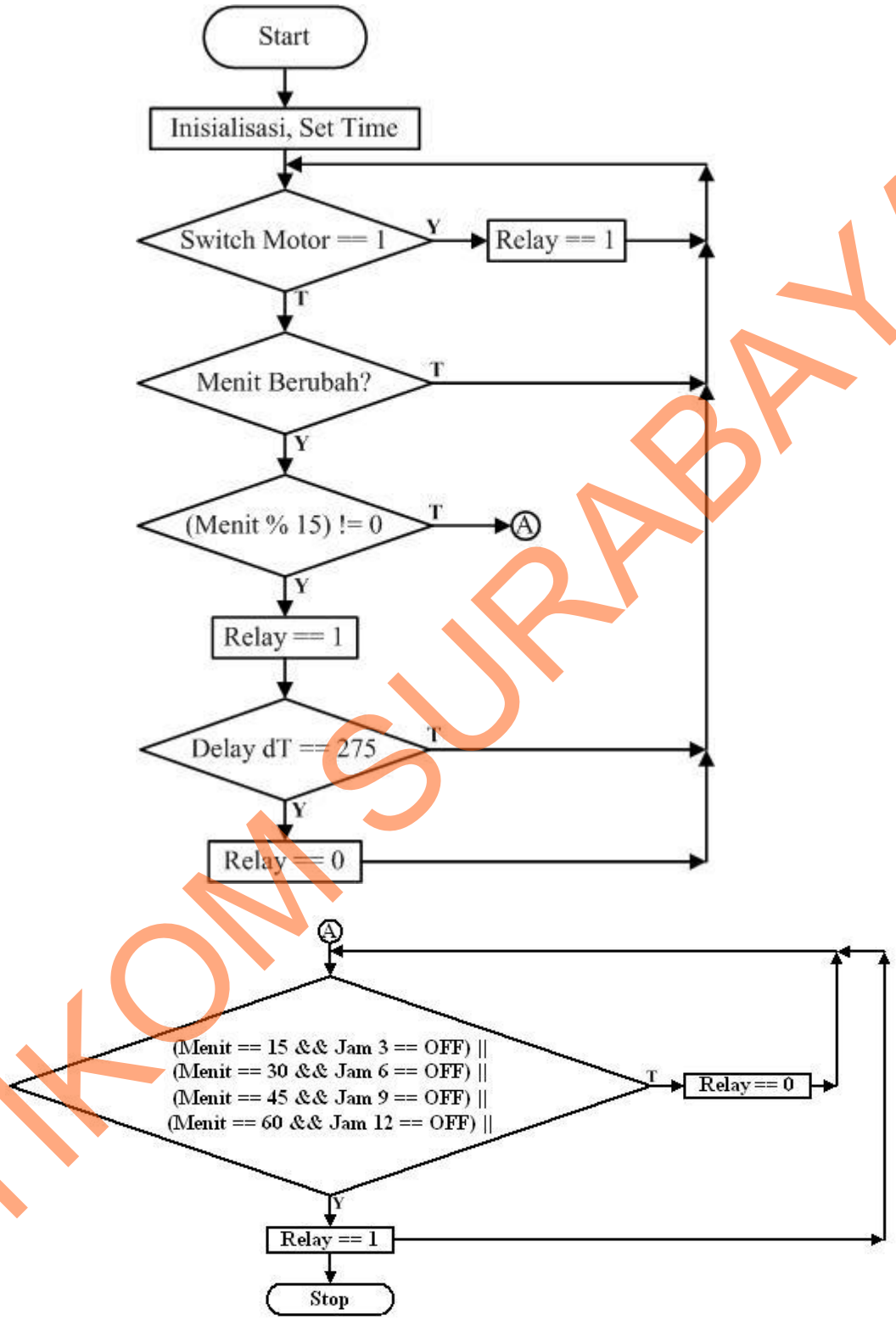
### 4.2.1 Flowchart

Flowchart perangkat lunak pada *microcontroller* terdapat pada Gambar 4.7 dan Gambar 4.8. Instruksi pertama yang dijalankan oleh *microcontroller* adalah initialisasi. Inisialisasi ini melingkupi register-register pada *microcontroller*, register-register pada RTC, dan variable-variabel yang akan digunakan pada program. Instruksi yang berikutnya akan dijalankan adalah pendeteksian penekanan tombol yang dilakukan *user*.

Bila *user* melakukan penekanan switch maka *relay* akan memicu motor untuk bergerak, hal ini bertujuan apabila waktu yang ditunjukkan oleh LCD sistem tidak menunjukkan waktu yang sama pada alat yang sebenarnya. Motor akan berhenti apabila *user* menghentikan penekanan switch dan waktu pada alat yang sebenarnya telah menunjukkan waktu yang sama dengan LCD sistem.

Setelah melakukan instruksi inisialisasi *user* tidak melakukan penekanan switch, maka program akan menjalankan instruksi selanjutnya yaitu pendeteksian menit, ini dilakukan untuk mendeteksi perubahan setiap menitnya. Bila menit berubah, program akan mendeteksi kembali apa yang akan dijalankan, jika menit yang saat ini sedang berjalan telah dibagi dengan 15 dan memiliki sisa bagi atau tidak sama dengan 0, maka *relay* akan memicu motor untuk berpindah ke menit berikutnya.

Apabila kondisi yang terjadi berkebalikan dengan kondisi yang diharapkan (menit yang saat ini sedang berjalan telah dibagi dengan 15 dan tidak memiliki sisa bagi atau sama dengan 0), maka program akan melakukan pendeteksian kembali, pada kondisi ini program akan mendeteksi 4 kondisi sekaligus, apabila salah satu kondisi terpenuhi *relay* akan memicu motor untuk menunjuk waktu setiap 15 menit.




Gambar 4.7 Flowchart program pada *microcontroller*

Program utama pada *microcontroller* yang disusun berdasarkan flowchart pada Gambar 4.7 dan 4.8 adalah sebagai berikut:

```
// Declare your global variables here
        #define RELAY PORTD.5
        #define SWITCH PIND.7
        #define JAM3 PINA.2
        #define JAM6 PINA.5
        #define JAM9 PINC.0
        #define JAM12 PINC.3
        unsigned char jam, menit, detik;
        unsigned char omenit, odetik;
        unsigned int i,j;
        unsigned char sensor[13];
        const int dT = 275;
        const int S_OFF = 0;
        const int S_ON = 1;
        unsigned char sjam[3], smenit[3], sdetik[3];


void main(void)
{
      // Input/Output Ports initialization
      // Port A initialization
      // Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out
      Func2=Out Func1=Out Func0=Out
      // State7=0 State6=0 State5=0 State4=0 State3=0 State2=0
      State1=0 State0=0
      PORTA=0x00;
      DDRA=0xFF;

      // Port B initialization
      // Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
      Func1=In Func0=In
      // State7=T State6=T State5=T State4=T State3=T State2=T
      State1=T State0=T
      PORTB=0x00;
      DDRB=0x00;

      // Port C initialization
      // Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out
      Func2=Out Func1=Out Func0=Out
      // State7=0 State6=0 State5=0 State4=0 State3=0 State2=0
      State1=0 State0=0
      PORTC=0x00;
      DDRC=0xFF;
```

```
// Port D initialization
// Func7=In Func6=Out Func5=In Func4=In Func3=In Func2=In
Func1=Out Func0=Out
// State7=T State6=0 State5=T State4=T State3=T State2=T
State1=0 State0=0
PORTD=0x00;
DDRD=0xBC;        ///---------->>>> OUTPUT INPUT

// I2C Bus initialization
i2c_init();

// DS1307 Real Time Clock initialization
// Square wave output on pin SQW/OUT: On
// Square wave frequency: 1Hz
rtc_init(0,1,0);

// Alphanumeric LCD initialization
// Connections specified in the
// Project|Configure|C Compiler|Libraries|Alphanumeric LCD
menu:
// RS - PORTB Bit 0
// RD - PORTB Bit 1
// EN - PORTB Bit 2
// D4 - PORTB Bit 4
// D5 - PORTB Bit 5
// D6 - PORTB Bit 6
// D7 - PORTB Bit 7
// Characters/line: 8
lcd_init(8);

// Global enable interrupts
#asm("sei")
    //rtc_set_time (15,8,20);
    rtc_get_time(&jam, &menit, &detik);
    omenit = menit;
    odetik = detik;
    while (1)
    {
       // Place your code here
       if(SWITCH == 1)
       {
             RELAY = 0;
             i = 0;
             while((SWITCH == 1) && (i < 110))
             {
                   i++;
             }
```

```
		if(i > 100)
		{
			RELAY = 1;
			while(SWITCH == 1);
		}
		RELAY = 0;
		delay_ms(3000);
	}
	else
	{
		omenit = menit;
		rtc_get_time(&jam, &menit, &detik);
		itoa(jam, sjam); itoa(menit, smenit);
		itoa(detik, sdetik);

		lcd_clear();
		lcd_puts(sjam);
		lcd_putchar(':');
		lcd_puts(smenit);
		lcd_putchar(':');
		lcd_puts(sdetik);

		if(omenit != menit) //jika menitnya berubah
		{
			if((menit % 15) != 0)
			{
				RELAY = 1;
				delay_ms(dT);
				RELAY = 0;
				delay_ms(3000);
			}
			else if((menit == 15) && (JAM3 == S_OFF))
			{
				RELAY = 1;
				while(JAM3 == S_OFF);
				RELAY = 0;
				delay_ms(3000);
			}
			else if((menit == 30) && (JAM6 == S_OFF))
			{
				RELAY = 1;
				while(JAM6 == S_OFF);
				RELAY = 0;
				delay_ms(3000);
			}
			else if((menit == 45) && (JAM9 == S_OFF))
			{
				RELAY = 1;
				while(JAM9 == S_OFF);
				RELAY = 0;
				delay_ms(3000);
```

```
                        }
                        else if((menit == 0) && (JAM12 == S_OFF))
                        {
                              RELAY = 1;
                              while(JAM12 == S_OFF);
                              RELAY = 0;
                              delay_ms(3000);
                        }
                  }
                  delay_ms(100);
            }
      }
}
```

### 4.3 Hasil Pengujian Program Menggunakan Simulasi

Pengujian-pengujian yang dilakukan pada sistem adalah pengujian pengaksesan RTC oleh *microcontroller* dan keluaran yang dihasilkan.

#### 4.3.1 Pengujian Pengaksesan RTC Oleh *Microcontroller*

Pada pengujian ini memiliki tujuan untuk mengetahui apakah *microcontroller* dapat melakukan pengaksesan pada RTC DS1307 atau tidak, baik untuk mengubah data pada RTC maupun untuk membaca data dari RTC.

Langkah-langkah untuk melakukan pengujian pada RTC DS1307 adalah sebagai berikut:

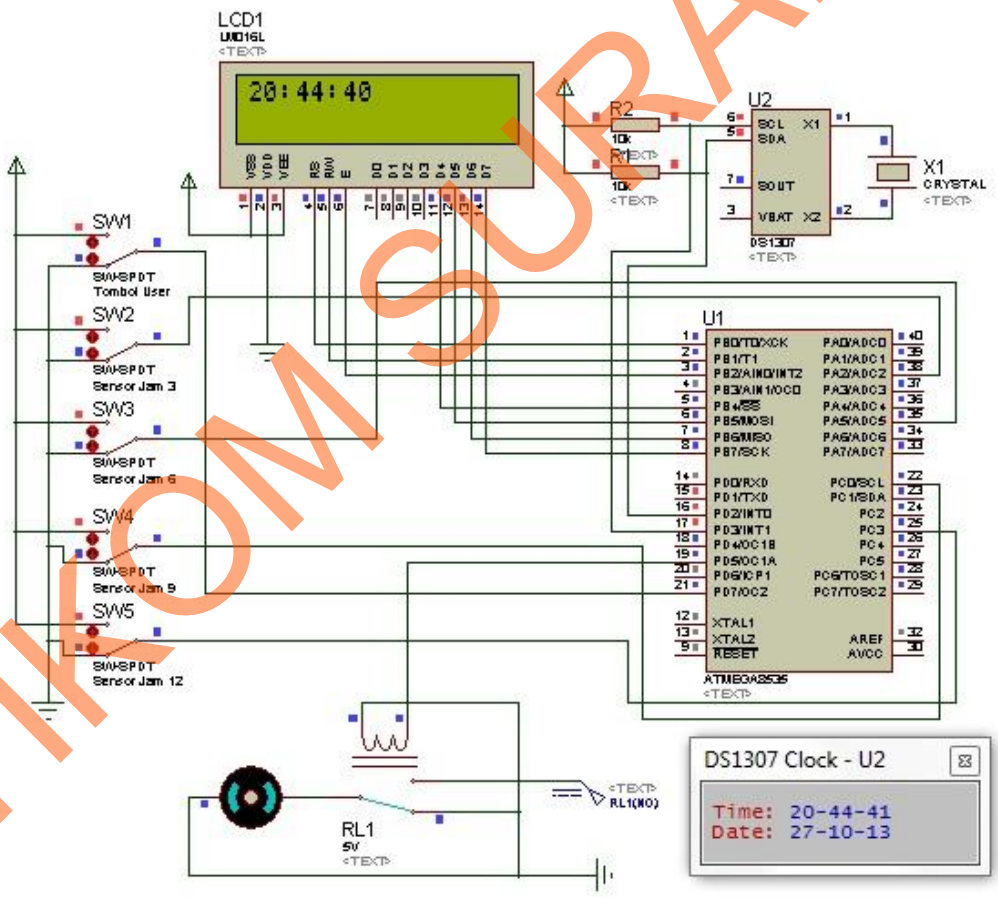
1. Berikan tegangan catu daya +9 V pada modul sistem, kemudian nyalakan sistem.
2. *Download* program pengujian RTC berikut pada *microcontroller*.

```
#include <mega8535.h>
#include <delay.h>

// Standard Input/Output functions
#include <stdio.h>
#include <stdlib.h>

// I2C Bus functions
#asm
   .equ __i2c_port=0x12 ;PORTD
   .equ __sda_bit=2
   .equ __scl_bit=3
#endasm
#include <i2c.h>

// DS1307 Real Time Clock functions
#include <ds1307.h>
void main(void)
{
    // USART initialization
    //Communication Parameters: 8 Data, 1 Stop, No
    Parity
    // USART Receiver: On
    // USART Transmitter: On
    // USART Mode: Asynchronous
    // USART Baud Rate: 9600
    UCSRA=0x00;
    UCSRB=0xD8;
    UCSRC=0x86;
    UBRRH=0x00;
    UBRRL=0x19;

    // I2C Bus initialization
    i2c_init();

    // DS1307 Real Time Clock initialization
    // Square wave output on pin SQW/OUT: On
    // Square wave frequency: 1Hz
    rtc_init(0,1,0);

    // Alphanumeric LCD initialization
    // Connections specified in the
    //Project|Configure|CCompiler|Libraries|Alphanumer
    ic   LCD menu:
    // RS - PORTB Bit 0
    // RD - PORTB Bit 1
    // EN - PORTB Bit 2
```

```
// D4 - PORTB Bit 4
// D5 - PORTB Bit 5
// D6 - PORTB Bit 6
// D7 - PORTB Bit 7
// Characters/line: 8
lcd_init(8);

// Global enable interrupts
#asm("sei")

   //Pengaturan waktu: 15:08:20
   rtc_set_time (15,8,20);
   rtc_get_time(&jam, &menit, &detik);

   omenit = menit;
   odetik = detik;

   while (1)
   {
      omenit = menit;

      rtc_get_time(&jam, &menit, &detik);
      itoa(jam, sjam); itoa(menit, smenit);
      itoa(detik, sdetik);

      lcd_clear();
      lcd_puts(sjam);
      lcd_putchar(':');
      lcd_puts(smenit);
      lcd_putchar(':');
      lcd_puts(sdetik);
   }
}
```

3. Matikan catu daya
4. Hubungkan DT-HiQ AVR USB ISP mkII dengan modul sistem dan komputer.
5. Nyalakan catu daya pada modul sistem, kemudian nyalakan sistem.
6. Amati pada LCD sistem, lakukan pengamatan data yang diterima dari RTC.

Dari langkah-langkah pengujian di atas didapatkan hasil bahwa RTC DS1307 dapat berjalan sesuai dengan harapan. Program yang di-*download* pada *microcontroller* pertama-tama akan melakukan inisialisasi register *control* pada RTC yang kemudian akan dilanjutkan dengan inisialisasi waktu yaitu jam, menit, dan detik. Setelah proses inisialisasi selesai, *microcontroller* akan melakukan pembacaan data pada RTC secara terus menerus yang kemudian akan dikirimkan pada LCD sistem sebagai output apabila terdapat perubahan data.




Gambar 4.8 Data waktu dari RTC yang dibaca oleh *microcontroller* dan ditampilkan pada LCD sistem