p-ISSN : 2746-7635
e-ISSN : 2808-5027

Modifikasi *Controlling Oil Level Hydraulic Pump Unit* (HPU) *Curing* Seyen *Plant* K Menggunakan RFID di PT. XYZ

Fauzan Malvin Satrio Hermawan[1)]
Jurusan Teknik Elektronika, Politeknik Gajah Tunggal
fauzanmalvin@gmail.com

Muhamad Wirdi[2)]
Jurusan Teknik Elektronika, Politeknik Gajah Tunggal
muhamadwirdi0@gmail.com

Adik Susilo Wardoyo[3)]
Jurusan Teknik Elektronika, Politeknik Gajah Tunggal
adiksusilo@poltek-gt.ac.id

ABSTRAK

Kasus yang marak terjadi pada sistem *reset alarm Oil Level Low Hydraulic Pump Unit* (HPU) yang masih bersifat *low security*, mengakibatkan banyak pihak yang dapat mengontrol diluar tanggungjawabnya, sehingga penulis membuat sistem *reset alarm Oil Level Low* HPU berbasis *Microcontroller Radio Frequency Identification* (RFID), Arduino Uno dan aplikasi penambahan data *member* baru menggunakan Visual Studio, bertujuan untuk membuat sistem *reset Oil Level* HPU menjadi lebih ketat keamanan sistemnya karena hanya dapat dikendalikan oleh orang tertentu saja.Metode yang akan digunakan dalam Modifikasi *Controlling Oil Level* HPU Berbasis *Microcontroller* RFID dan Arduino Uno, terdiri dari beberapa tahap yaitu identifikasi dan analisa kebutuhan, perancangan perangkat lunak dan aplikasi penambahan *member* baru, perangkat keras, pembuatan alat, dan pengujian alat, perangkat keras terdiri dari sistem Arduino Uno sebagai pengendali utama, *chip id card* sebagai media pengendalian, RFID sebagai media penghubung.Berdasarkan hasil pengujian disimpulkan bahwa Modifikasi sistem *reset Controlling Oil Level* HPU menggunakan RFID layak dibuat sebagai kontrol pada *Oil Level* HPU, dan aplikasi penambah *member* baru layak digunakan. Arduino Uno dan aplikasi ini dapat bekerja sesuai dengan prinsip kerja yang telah dirancang, serta dapat mempermudah pihak yang bertanggungjawab.

Kata Kunci : *HPU, Reset Alarm, Microcontroller, Aplikasi penambah member baru*

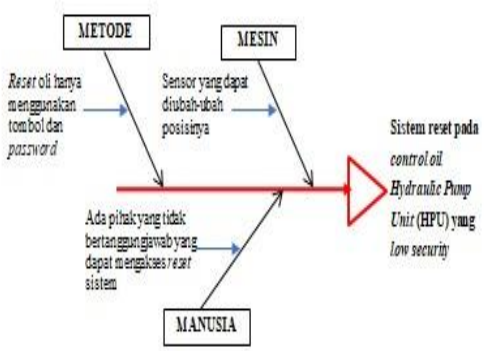
I. PENDAHULUAN

Berkembangnya dunia teknologi yang sangat cepat, menuntut manusia untuk menciptakan teknologi yang dapat menunjang pekerjaan manusia menjadi lebih mudah, dari pekerjaan yang membutuhkan mobilitas dan tenaga besar menjadi lebih ringan dengan adanya otomatisasi yang tercipta dari teknologi. Teknologi di era *modern* ini menjadi hal yang sangat penting didalam perindustrian, berbagai macam teknologi telah lahir mulai dari teknologi yang bersifat pengembangan dari yang sebelumnya (modifikasi) hingga teknologi yang belum ada sebelumnya. Dengan adanya pertumbuhan teknologi yang pesat 10 tahun terakhir khususnya, meningkatkan persaingan antar perusahaan untuk selalu berupaya meningkatkan teknologi yang ada di perusahaannya dengan membuat inovasi baru [1].

Dalam melakukan inovasi teknologi, perusahaan pasti selalu mempertimbangkan kualitas, kuantitas, dan kepuasan konsumen apakah masih terjamin atau tidak, dengan mempertimbangkan *cost* yang digunakan untuk upaya inovasi seminimal mungkin namun tetap menjamin kualitas dan kuantitas produk. Di perusahaan PT. XYZ contohnya, selalu berupaya untuk meningkatkan teknologi yang ada, dengan membentuk tim *improvement* di setiap *plant*.

PT. XYZ juga selalu meningkatkan teknologi keamanan yang ada pada mesin, baik itu keamanan untuk *operator* mesin atau keamanan yang mempengaruhi *durrability* mesin. Penerapan teknologi pada keamanan mesin pada tangki *Hydraulic Pump Unit* (HPU) contohnya. PT. XYZ menggunakan sensor KQ-6001 yang dapat membaca keberadaan oli yang ada pada tangki, dimana oli didalam tangki ini berperan dalam operasional produksi *curing* untuk menggerakan *cylinder hydraulic* pada proses *Curing*, sensor KQ-6001 di pasang pada batas bawah yang akan memicu *alarm* ketika batas bawah *level* oli terlewat dari permukaan oli, dengan ini menandakan tangki oli dalam keadaan *low* dan oli harus di isi kembali.

Ketika *alarm Oil Level Low* menyala, hal yang perlu dilakukan adalah melakukan pengisian oli pada tangki selain itu tim *engineering* juga harus mengecek adanya kebocoran atau tidak, baik itu pada tangki, *cylinder hydraulic*, *manifold* dan selang penghubung oli, jika dirasa sudah tidak adanya kebocoran maka *alarm Oil Level Low* di *reset* dengan menggunakan *password*, dimana *password* yang digunakan sama di semua mesin, di PT. XYZ sendiri jumlah mesin ada 126 mesin, sehingga sangat menguras tenaga dan waktu apabila mengganti *password* ditambah lagi belum adanya aplikasi yang mempermudah penggantian *password* sehingga harus memrogram menggunakan ladder diagram pada CX Developer di setiap mesin, dan banyak pihak yang tidak bertanggungjawab dapat mengakses *reset alarm* tersebut, dan mesin bisa dikerjakan kembali tanpa adanya pengisian dan pengecekan kebocoran terlebih dahulu, hal ini dapat mengakibatkan *top up oli over*, karena tangki oli lebih sedikit dari batas minimal, bahkan hal terburuknya dapat merusak pompa dan motor. Berdasarkan pemaparan tersebut itulah penulis membuat modifikasi sistem keamanan *reset alarm Oil Level Low* dengan menggunakan RFID dan juga Aplikasi Penambahan data baru yang memudahkan tim *engineering* dalam menambah *member* baru penanggungjawab mesin, dimana *id card* pihak *engineering* saja yang dapat mengakses *reset alarm* tersebut.



Gambar 1. *Fishbone*

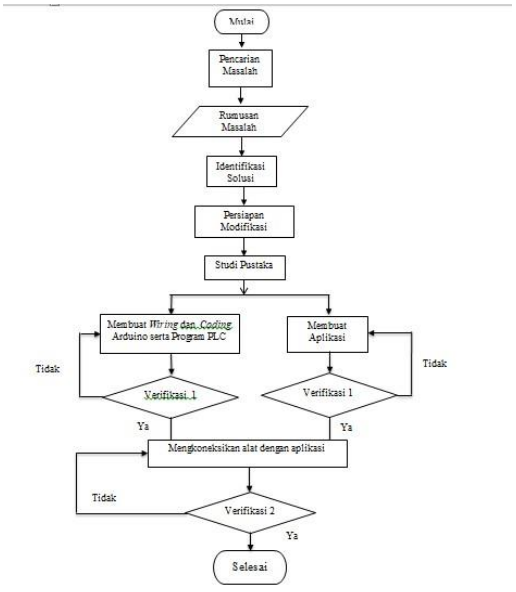
Berdasarkan analisis permasalahan tersebut, perlu adanya modifikasi pada sistem *reset Control Oil* HPU yang *low security*. Modifikasi alat tersebut dilakukan atas dasar beberapa unsur, yaitu unsur manusia, mesin dan metode yang ada pada sistem *reset alarm Oil Level Low* HPU. Dimana pada kondisi saat ini, sistem *reset alarm* hanya menggunakan *push button* dan *password* saja dan *password* yang digunakan sama di 126 mesin sehingga membuat keamanan ini masih bersifat *low security*. Dengan kondisi saat ini, perlu dilakukan modifikasi *reset alarm Oil Level Low* HPU dengan berbagai unsurnya.

Sistem kontrol elektrik pada modifikasi ini menggunakan *microcontroller* yaitu Arduino Uno. Nantinya, Arduino Uno akan menerima sinyal dari RFID kemudian akan diolah untuk mnegaktifkan relay. Relay disini digunakan untuk memberi *inputan* ke PLC (*Programmable Logic Controller*).

Microcontroller adalah sistem *microprocessor* lengkap yang ada didalam *chip*, *microcontroller* berbeda dengan *microprocessor* yang ada pada *Personal Computer* (PC), dikarenakan yang ada pada sebuah *microcontroller* umumnya sudah berisi komponen pendukung sistem minimum *microprocessor*, yaitu antarmuka *Input*/*Output* (I/O) dan memori, sedangkan yang ada di dalam *microprocessor* umumnya berisi *Central Processing Unit* (CPU) saja [2]. Arduino adalah *platform hardware* terbuka, jadi siapapun yang berkeinginan membuat alat elektronik yang interaktif dengan *software* dan *hardware* yang mudah digunakan serta fleksibel. *Microcontroller* ini pemrogramannya hampir mirip dengan bahasa pemrograman C yang dinamakan bahasa pemrograman Arduino. Atmel merilis basis Arduino menggunakan *microcontroller* ATMega, tetapi ada juga perusahaan atau individu dengan menggunakan *microcontroller* lain selain ATMega, namun tetap kompatibel dengan Arduino *level hardware*. Pada tahun 2005 di Italia terdapat proyek berawalnya Arduino, sampai 2010 terjual lebih dari 120.000 unit, dinamakan Arduino karena memiliki arti teman yang kuat, nama ini diambil dari sebuah nama maskulin. *Platform* Arduino yaitu *shield*, Arduino *board*, Arduino *development environment* dan bahasa pemrograman Arduino, *shield* yaitu suatu papan yang bisa dipasang diatas Arduino *board* agar menambah kemampuan Arduino *board* itu sendiri, sedangkan Arduino *development environment* yaitu sebuah perangkat lunak untuk meng-c*ompile* dan menulis program Arduino, digunakan juga untuk men-*transfer* program yang telah di-c*ompile* ke penyimpanan program Arduino [3]. RFID adalah sebuah teknologi berbasis gelombang radio yang mampu mengidentifikasi objek dengan simulatan dengan jarak yang dekat. Sebagai pengganti *barcode* dikembangkanlah suatu alat yang bernama RFID. Terdapat RFID *tag* dan RFID *reader* yang digunakan untuk melakukan proses identifikasi. RFID *tag* ditempelkan di sebuah objek yang diidentifikasi, RFID ini mempunyai data berupa angka-angka unik. Salah satu keunikannya yaitu setiap RFID *tag* berbeda-beda angkanya tidak akan ada yang sama. RFID *reader* yaitu digunakan sebagai pembaca RFID *tag* yang mana suatu objek tersebut dapat dilakukan pengidentifikasian [4]. PLC yaitu suatu alat elektronik yang berjalan secara digital dan dibuat untuk dipakai pada lingkungan industri. Sistem ini menggunakan memori sebagai penyimpan perintah-perintah dan sebagai pengimplementasian fungsi-fungsi spesifik, contohnya seperti logika, sekuensial, aritmatika, pencacahan, dan perwaktuan untuk mengontrol suatu mesin atau suatu proses di modul *Input*/*Output* analog atau juga digital [5].Microsoft Visual Studio adalah suatu *Integrated Development Environment* (IDE) yang dibuat oleh Microsoft yang bertujuan untuk mengembangkan aplikasi *console*, aplikasi yang ada di aplikasi web maupun di sistem operasi windows. Visual Studio ini cukup lengkap dalam bahasa pemrogramannya sehingga seorang programmer dapat memilih banyak bahasa yang akan digunakan, terdapat banyak bahasa yaitu Visual C++, Visual Basic, Visual C#, Visual Basic.Net, Visual J++, Visual InterDev, Visual FoxPro serta Visual Source Safe [6].

II. METODE PENELITIAN

Berikut Pada bagian ini akan dipaparkan bagaimana alur penelitian yang dilakukan dalam penelitian saat ini :



Gambar 2. *Flow Chart* Alur Penelitian

1. Pencarian Masalah

   Pada proses ini, kami mencari permasalahan yang ada pada proses *curing Plant* K di PT. XYZ.
2. Rumusan Masalah

Pada proses ini, kami mulai menemukan masalah yang ada pada proses *curing*, kami mendapati bahwa terdapat keamanan yang kurang pada sistem *reset alarm Oil Level Low* HPU, sehinga hal ini bersifat *low security*.

3. Identifikasi Solusi

Pada proses ini kami melakukan pencarian solusi yang tepat untuk permasalahan tersebut, setelah melakukan diskusi kami mendapatkan ide yaitu menambahkan sistem keamanan *reset alarm Oil Level Low* HPU dengan menambahkan RFID sebagai *reset* tambahan, menjadikan keamanan pada *reset* oli menjadi keamanan ganda.

4. Persiapan Modifikasi

Pada tahap ini, kami melakukan persiapan modifikasi sistem tersebut yaitu mengubah sistem keamanan *reset alarm Oil Level Low* HPU dengan RFID sebagai *reset*.

5. Studi Pustaka

Pada tahap ini, dilakukakn studi terhadap aktivitas-aktivitas yang sama serta mencari teori yang berkaitan dengan penelitian yang sedang dilakukan melalui buku, artikel ilmiah serta jurnal sebagai referensi.

6. Membuat Wiring dan Coding Arduino, serta Membuat Program PLC dan Aplikasi

Pada tahap ini, dilakukan pembuataan *wiring* dan koding program Arduino/*source code*. Pa proses ini, juga dilakukan pemodifikasian *ladder diagra*m pada PLC yang sudah dibuat sebelumnya, pada tahap ini juga dilakukan pembuatan aplikasi, aplikasi ini bertujuan untuk menambahkan *member* baru atau *id card* baru.

7. Verifikasi 1

Pada tahap ini, dilakukan pengujian terhadap apa yang sudah dibuat. Apakah hasilnya sesuai dengan spesifikasi yang diinginkan atau belum, jika masih belum dan terjadi kesalahan maka dilakukan kembali ke tahap sebelumnya.

8. Mengkoneksikan Alat dan Aplikasi

Pada tahap ini, kami menghubungkan atau mengoneksikan Arduino dengan aplikasi yang dibuat.

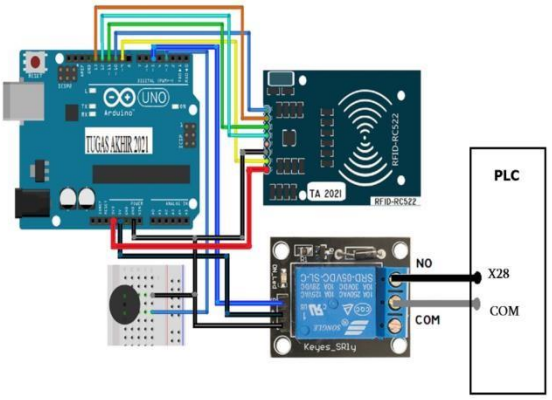
9. Verifikasi 2

Pada tahap ini, dilakukan pengujian terhadap apa yang sudah dibuat, apakah hasilnya sesuai dengan spesifikasi yang diinginkan atau belum, jika masih belum dan terjadi kesalahan maka dilakukan kembali ke tahap sebelumnya, jika sudah tidak ada kesalahan dan sudah sesuai dengan spesifikasi yang diinginkan maka pembuatan alat tersebut selesai dan dapat digunakan.

III. HASIL DAN PEMBAHASAN

A. Gambar Rangkaian

Pada pembahasan ini akan membahas tentang gambar rangkaian alat yang dibuat.Gambar 3 merupakan gambar rangkaian dari alat yang dibuat secara keseluruhan.



Gambar 3. Gambar Rangkaian

Gambar 3 adalah rangkaian dari alat yang dibuat, dimana terdapat Arduino Uno, RFID, *buzzer*, relay dan PLC. Relay mendapat tegangan 5 volt dari Arduino sedangkan RFID mendapatkan tegangan 3 volt, selanjutnya pada *buzzer* terdapat 2 kaki, kaki yang pertama masuk ke pin GND dari arduino dan kaki yang kedua masuk ke pin 5 Arduino, kemudian pada relay dihubungkan ke pin 3 volt, GND dan pin 5 dari Arduino lalu *output* dari relay dihubungkan ke pin X28 dan COM dari PLC. Selanjutnya pin-pin pada RFID dihubungkan ke Arduino sesuai gambar diatas.

B. Program Arduino

1. Void Setup

Pada pembahasan ini akan membahas tentang program *void setup* pada Arduino. Gambar 4 adalah gambar dari program *void setup*.

```
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  rfidBegin();
  pinMode(relay, OUTPUT);
  pinMode(Buzzer, OUTPUT);
  digitalWrite(relay, LOW);
  digitalWrite(Buzzer, LOW);
}
```

```
        digitalWrite(relay, HIGH);
        delay(1000);
        digitalWrite(relay, LOW);
      }
      else {
        // kalo data tidak ada di eeprom
        TAG="";
        digitalWrite(relay, LOW);
        delay(10);
      }
    }
  }
```

Gambar 1. Program *Void Setup*

Gambar 4 adalah program *void setup* yang mana isinya yaitu ada *Serial.begin* (9600) berarti bahwa *serial* yang digunakan adalah baut 9600, kedua ada *pinMode* (*relay,OUPUT*) artinya bahwa *relay* adalah sebuah keluaran dari Arduino selanjutnya ada *pinMode* (*Buzzer,OUTPUT*) berarti bahwa *buzzer* adalah keluaran dari Arduino, ketiga ada *digitalWrite (relay,LOW)* artinya bahwa *relay* pada saat awal dalam kondisi *LOW* atau mati kemudian *digitalWrite (Buzzer,LOW)* artinya bahwa kondisi awal dari *Buzzer* adalah *LOW* atau mati.

2. Void Loop

Pada pembahasan ini akan membahas tentang program *void loop* pada Arduino. Gambar 5 adalah gambar dari program *void loop*.

```
void loop() {
  // put your main code here, to run repeatedly:
  while(!Serial){
    modeNormal();
  }
  modeEdit();
}
```

Gambar 5. Program *Void Loop*

Gambar 5 merupakan program *void loop* di Arduino dimana nantinya *void loop* ini adalah yang dilakukan untuk menjalankan fungsi secara berulang, program *void loop* di isi oleh *mode* normal dan *mode edit* dimana *mode* normal dan *edit* akan dibuat kodingan tersendiri yaitu *void* modeNormal dan *void* modeEdit, v*oid* tersebut akan berfungsi bergantian sesuai dengan kodingan yang dibuat.

3. *Void Mode* Normal

Pada pembahasan ini akan membahas tentang program *void* modeNormal pada Arduino. Gambar 8 adalah gambar dari program *void Mode* Normal.

Gambar 6. Program *Void Mode* Normal

Gambar 6 menunjukkan program dari *void* modeNormal yang berarti bahwa alat berjalan sesuai fungsinya, *Void* ini dijalankan oleh *void* loop yang sudah dibahas sebelumnya, v*oid* modeNomal berisi tentang fungsi dari RFID untuk menjalankan *relay*. Jadi, ketika pengguna melakukan *tag* pada RFID maka RFID dan Arduino akan membaca data yang tersimpan, apabila kartu tersebut terdaftar maka program *digitalWrite (relay,HIGH)* berjalan yang artinya *relay* hidup setelah itu ada *delay* (1000) artinya bahwa *relay* hidup selama 1 detik dan selanjutnya *digitalWrite (relay,LOW)* yang berarti *relay* mati, kemudian jika data kartu tidak terdaftar maka relay akan tetap mati yaitu dengan koding digitalWrite (relay,LOW).

4. *Void Mode Edit*

Pada pembahasan ini akan membahas tentang program *void* modeEdit pada Arduino. Gambar 7 adalah gambar dari program *void* modeEdit.

```
void modeEdit() {
  String tombol, id, perintah, nama;
  int idInt;
  digitalWrite(Buzzer, HIGH);
  getTAG();
  if (TAG != "") {
    tambahData(10, TAG); // memori sementara
    TAG="";
  }

    // dari aplikasi "2readdt"
    tombol = Serial.readString();
    id = tombol.substring(0, 1); // charachter pertama
    idInt = id.toInt();
    perintah = tombol.substring(1,7); // charachter 2 sampe 6
    nama = tombol.substring(7);
```

Gambar 7.Program *Void Mode Edit*

Gambar 7 adalah program dari *void* modeEdit yang artinya alat dalam kondisi mengedit, *Void* modeEdit dijalankan oleh *void* loop yang sudah dijelaskan sebelumnya, ketika pada aplikasi *button mode edit* ditekan maka program *mode edit* ini akan berfungsi. Jadi, pengguna akan melakukan *tag* RFID dan data dari kartu akan terbaca serta tersimpan pada EEPROM 10, kemudian program diatas adalah untuk membaca teks yang dikirimkan dari aplikasi, Arduino akan membaca angka dan huruf yang dikirimkan dimana angka akan terbaca pada baris pertama dan huruf akan terbaca pada baris 2 sampai 6 contohya, yaitu “4readdt” artinya “4” adalah angka yang dibaca baris pertama dan

"readdt" adalah huruf dari baris 2 sampai 6. Gambar 8 adalah lanjutan dari *void* modeEdit.

Gambar 8 menunjukkan kelanjutan koding dari *void* modeEdit yang berisi tentang fungsi dari tombol-tombol pada aplikasi. Pertama ada if (perintah == "readdt") yaitu program dari tombol *read* 1 dan *read* 2 pada aplikasi yang artinya ketika tombol ditekan akan mengirimkan teks "readdt" kemudian Arduino akan menerima teks tersebut untuk menjalankan koding tersebut dan data akan dibaca serta disimpan pada EEPROM 10. Kedua ada else if (perintah == "tambah") program ini digunakan untuk tombol tambah pada aplikasi yang mana ketika tombol tambah ditekan maka aplikasi akan mengitimkan teks tambah dan akan diterima oleh Arduino kemudian Arduino akan menjalankan program tersebut yaitu akan menyimpan data dari EEPROM 10 ke EEPROM *tag*. EEPROM tag ini berarti data akan tersimpan tetap dimana data yang tersimpan adalah nama dan nomor *id card* dari kartu *member* atau pengguna.

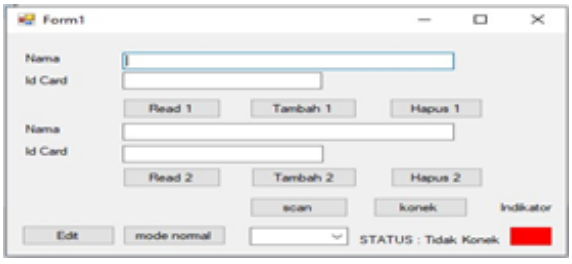
```
if (perintah == "readdt") {
  Serial.println(bacaData(10)); //yang dikirim id sama no kartu
}

}
else if (perintah == "tambah") {
  String tag = bacaData(10);
  tambahData(idInt, tag);
  tambahData(idInt+10, nama);

}
```

Gambar 8. Program *Void Mode Edit*

C. Aplikasi

Pada pembahasan ini akan membahas tentang aplikasi yang digunakan untuk menambahkan *member*/*id card* baru. Gambar dibawah ini adalah tampilan awal dari aplikasi.

Terdapat beberapa fitur dalam aplikasi ini yaitu tabel nama, tabel *id card*, *button read* 1, *button read* 2, *button* tambah 1, *button* tambah 2, *button* hapus 1, *button* hapus 2, *button scan*, *button* konek, *button* tidak konek, *button* edit, *button mode* normal, indikator, dan Status.



Gambar 9. Tampilan Aplikasi

Penjelasan dari fitur-fitur tersebut,sebagai berikut :

1. Scan

Fitur *scan* ini digunakan untuk membaca COM *port* dari Arduino yang akan digunakan, COM akan muncul pada *Listbox* dibawah fitur *scan* kemudian pilih COM yang terbaca pada aplikasi.

2. COM

Fitur COM digunakan untuk menampilkan COM yang terbaca oleh aplikasi.

3. Konek

Fitur konek digunakan untuk mengkoneksikan aplikasi dengan Arduino, fitur konek dapat digunakan setelah pengguna memilih COM yang digunakan.

4. Indikator

Indikator digunakan untuk menandakan aplikasi terkoneksi atau tidak terkoneksi dengan Arduino, jika terkoneksi dengan Arduino maka indikator akan berubah warna menjadi warna hijau tetapi jika Arduino tidak terkoneksi dengan Arduino maka indikator berwarna merah.

5. Status

Status digunakan untuk menandakan aplikasi terkoneksi atau tidak terkoneksi dengan Arduino, jika terkoneksi dengan Arduino maka indikator akan berubah kata menjadi terkoneksi tetapi jika Arduino tidak terkoneksi dengan Arduino maka Status tidak terkoneksi.

6. Edit

Pada fitur *edit* digunakan untuk mengubah *mode* pada Arduino menjadi *mode edit*, *mode edit* ini adalah *mode* yang digunakan untuk menjalankan fitur *read* dan fitur tambah.

7. Mode Normal

Mode Normal digunakan untuk menjalankan alat agar berjalan seperti semula atau sesuai fungsi semulanya setelah pengguna mengedit alat tersebut yaitu menambahkan atau menghapus *id card.*

8. Read

Tombol *Read* digunakan untuk membaca *id card* baru yang akan ditambahkan yaitu dengan cara pengguna *tap id card* pada *reader* RFID setelah itu *klik* fitur r*ead* 1 atau *read* 2, nantinya angka dari *id card* tersebut akan muncul pada tabel *id card*. Terdapat 2 tombol *read* pada aplikasi ini fungsinya sama tetapi penggunaannya berbeda yaitu tombol *read* yang pertama digunakan untuk membaca data baru dari *member* pertama dan tombol *read* yang kedua digunakan untuk membaca data baru dari *member* kedua.

9. Tambah

Tombol tambah berfungsi untuk menambahkan *id card* baru setelah *id card* terbaca oleh aplikasi atau setelah *klik* tombol *read*.

Terdapat 2 tombol tambah pada aplikasi ini fungsinya sama tetapi penggunaannya berbeda yaitu tombol tambah yang pertama digunakan untuk menambahkan data baru dari *member* pertama dan tombol tambah yang kedua digunakan untuk menambahkan data baru dari *member* kedua.

10. Hapus

Tombol hapus digunakan untuk menghapus data *member*. Terdapat 2 tombol hapus pada aplikasi ini fungsinya sama tetapi penggunaannya berbeda yaitu tombol hapus yang pertama digunakan untuk menghapus data dari *member* pertama dan tombol hapus yang kedua digunakan untuk menghapus data dari *member* kedua.

11. Tabel Nama

Tabel nama memiliki 2 fungsi yaitu pertama untuk menuliskan nama untuk *member* baru, terdapat 2 tabel nama pada aplikasi ini fungsinya sama tetapi penggunaannya berbeda yaitu tabel nama yang pertama digunakan untuk menuliskan nama dari *member* pertama dan tabel nama yang kedua digunakan untuk menuliskan nama dari *member* kedua.

12. Tabel Id Card

Tabel *id card* digunakan untuk menampilkan angka atau kode yang terdapat pada *chip id card*, tabel ini dapat menampilkan data baru, terdapat dua tabel *id card* yaitu tabel yang pertama digunakan untuk menampilkan kode dari *member* pertama dan tabel yang kedua digunakan untuk menampilkan kode dari *member* kedua.

D. Program Visual Studio

1. *Button* Konek

Pada pembahasan ini akan membahas tentang program dari *button* konek. Gambar 10 adalah gambar dari program *button* konek.

```
Private Sub konek_Click(sender As System.Object, e As System.EventArgs) Handles konek.Click
    Try
        SerialPort1.BaudRate = "9600"
        SerialPort1.PortName = ComboBox2.SelectedItem
        SerialPort1.Open()
        Timer1.Start()
        Label1.Text = "STATUS: Konek"
        PictureBox1.BackColor = Color.Green
        TidakKonek.Enabled = True
        TidakKonek.BringToFront()
    Catch ex As Exception
        MsgBox("Please check the hardware, COM, Baud Rate and try again", MsgBoxStyle.Critical, "Connection failed
    End Try


End Sub
```

Gambar 10.Program *Button* Konek

Program ini berfungsi untuk mengkoneksikan Arduino dengan aplikasi, ketika *button* konek di-*klik* maka *serial port* langsung terhubung pada *serial port* 9600 pada Arduino, selanjutnya program ini juga berfungsi ketika *button* konek diklik maka status akan berubah menjadi konek dan indikator konek akan berubah menjadi warna hijau. Jika terjadi kesalahan maka akan muncul peringatan "*Please Check the hardware ,COM, Baut Rate and Try again*" artinya antara *hardware* dan aplikasi belum terhubung atau ada kesalahan dan pengguna harus mengecek alat yang digunakan.

2. *Button Scan*

Pada pembahasan ini akan membahas tentang program dari *button scan*. Gambar 11 adalah gambar dari program *button scan*.

```
Private Sub scan_Click(sender As System.Object, e As System.EventArgs) Handles scan.Click
    Try
        If Label1.Text = "STATUS: Konek" Then
            MsgBox("Koneksi sudah ada", MsgBoxStyle.Critical, "Warning!!!")
        End If
        ComboBox2.Items.Clear()
        Dim myPort As Array
        Dim i As Integer

        myPort = IO.Ports.SerialPort.GetPortNames()
        ComboBox2.Items.AddRange(myPort)
        i = ComboBox2.Items.Count
        i = i - i

        ComboBox2.SelectedIndex = i

    Catch ex As Exception
        MsgBox("COM PORT Tidak Terdetek", MsgBoxStyle.Critical, "Warning!!!")
        ComboBox2.Text = ""
        ComboBox2.Items.Clear()
    End Try


End Sub
```

Gambar 11. Program *Button Scan*

Program diatas berfungsi untuk membaca COM yang terhubung dengan Arduino. Cara kerjanya jika *button scan* diklik maka apabila status sudah konek akan muncul pesan "Koneksi sudah ada" dan nantinya akan muncul COM pada *Combobox*. Jika COM tidak terdeksi maka akan muncul peringatan "COM *PORT* tidak terdeteksi".

3. *Form*

Pada pembahasan ini akan membahas tentang program dari tampilan aplikasi. Gambar 12 adalah gambar dari program tampilan aplikasi.

```
Private Sub Form1_Load(sender As System.Object, e As System.EventArgs) Handles MyBase.Load
    Me.CenterToParent()
    TidakKonek.Enabled = False
    TidakKonek.SendToBack()
    konek.Enabled = True
    konek.BringToFront()
    Label1.Text = "STATUS : Tidak Konek"
    PictureBox1.Visible = True
    PictureBox1.BackColor = Color.Red

End Sub
```

Gambar 12. Program *Form*

Program diatas berfungsi untuk program tampilan awal dari aplikasi, jadi pada tampilan awal aplikasi *button* konek akan muncul terlebih dahulu daripada *button* tidak konek, *button* tidak konek terletak dibalik atau *sendtoback* dari *button* konek. Selanjutnya status pada awal tampilan bertuliskan tidak konek dan indikator berwarna merah.

4. *Button* Tidak Konek

Pada pembahasan ini akan membahas tentang program dari *button* tidak konek. Gambar 13 adalah gambar dari program *button* tidak konek.

```
Private Sub Form1_Load(sender As System.Object, e As System.EventArgs) Handles MyBase.Load
    Me.CenterToParent()
    TidakKonek.Enabled = False
    TidakKonek.SendToBack()
    konek.Enabled = True
    konek.BringToFront()
    Label1.Text = "STATUS : Tidak Konek"
    PictureBox1.Visible = True
    PictureBox1.BackColor = Color.Red

End Sub
```

Gambar 13.Program Tidak Konek

Program di atas digunakan untuk memutus koneksi Arduino dengan aplikasi. Cara kerjanya yaitu apabila *button* tidak konek diklik maka status akan berubah menjadi Tidak Konek dan indikator akan berubah menjadi warna merah. Pada kondisi ini juga *button* tidak konek akan bertukar menjadi *button* konek karna pada awal tampilan aplikasi button tidak konek terletak dibalik *button* konek.

5. *Button* Edit

Pada pembahasan ini akan membahas tentang program dari *button* edit. Gambar 14 adalah gambar dari program *button* edit.

```
Private Sub Edit_Click(sender As System.Object, e As System.EventArgs) Handles Edit.Click
    Try
        Dim text As String
        text = "edit"
        SerialPort1.Write(text)
        MsgBox("Anda dalam mode programmer", MsgBoxStyle.Information, "Pesan")
    Catch ex As Exception
        MsgBox("error " + ex.Message, MsgBoxStyle.Critical, "Warning!!!")
    End Try

End Sub
```

Gambar 14. Program *Button* Edit

Program diatas digunakan untuk merubah program pada *mode edit.* Cara kerjanya apabila *button edit* di klik maka akan mengirimkan teks "edit" ke Arduino dan akan muncul pesan "Anda dalam *mode programmer*".Pada Arduino *mode edit* ini diprogam untuk membaca *id card* baru serta penambahan data *member* baru. Apabila terjadi kesalahan atau *error* maka akan ada peringatan "*Error*".

6. *Button* Mode Normal

Pada pembahasan ini akan membahas tentang program dari *button mode* normal. Gambar 15 adalah gambar dari program *button mode* normal.

```
Private Sub modenormal_Click(sender As System.Object, e As System.EventArgs) Handles modenormal.Click
    Try
        Dim text As String
        text = "normal"
        SerialPort1.Write(text)
        MsgBox("Anda dalam mode normal", MsgBoxStyle.Information, "Pesan")
    Catch ex As Exception
        MsgBox("error " + ex.Message, MsgBoxStyle.Critical, "Warning!!!")
    End Try

End Sub
```

Gambar 15. Program *Button* Mode Normal

Program diatas digunakan untuk merubah program menjadi *mode* normal. Apabila *button mode* normal diklik maka aplikasi akan mengirimkan teks "normal" ke Arduino dimana pada program Arduino *mode* normal berfungsi untuk menjalankan alat seperti semula atau sesuai dengan fungsinya. Jika *button mode* normal tidak ada kesalahan maka akan muncul pesan "Anda dalam *mode* normal" tetapi, jika terdapat keesalahanmaka akan muncul peringatan "*error*".

7. *Button* Tambah 1

Pada pembahasan ini akan membahas tentang program dari *button* tambah 1. Gambar 16 adalah gambar dari program *button* tambah 1.

```
Private Sub tambah_Click(sender As System.Object, e As System.EventArgs) Handles tambah.Click
    Try
        Dim text As String
            text = "4tambah" + txnama.Text
            SerialPort1.Write(text)
            tx1.Text = ""
            txnama.Text = ""
            MsgBox("Data berhasil ditambahkan", MsgBoxStyle.Information, "Pesan")
    Catch ex As Exception
        MsgBox("error di tombol tambah1" + ex.Message, MsgBoxStyle.Critical, "Warning!!!")
    End Try


End Sub
```

Gambar 16. Program *Button* Tanbah 1

Program diatas merupakan program untuk menambahkan data baru *member* 1. Apabila *button* tambah 1 diklik maka aplikasi akan mengirimkan teks "4tambah" ke Arduino Uno dan nantinya pada Arduino akan menjalankan fungsinya, yang mana fungsi dari *button* tersebut untuk menambahkan data baru *member* 1 yang akan disimpan di EEPROM Arduino. Jika ada kesalahan maka akan muncul peringatan "*error di tombol tambah1*". Kesalahan yang ada salah satunya yaitu pengguna belum menekan tombol *edit* pada aplikasi.

8. *Button* Tambah 2

Pada pembahasan ini akan membahas tentang program dari *button* tambah 2. Gambar 17 adalah gambar dari program *button* tambah 2.

```
Private Sub tambah2_Click(sender As System.Object, e As System.EventArgs) Handles tambah2.Click
    Try
        Dim text As String
        text = "2tambah" + Textnama2.Text
            SerialPort1.Write(text)
            tx2.Text = ""
            Textnama2.Text = ""
            MsgBox("Data berhasil ditambah", MsgBoxStyle.Information, "Pesan")
    Catch ex As Exception
        MsgBox("error di tombol tambah2", MsgBoxStyle.Critical, "Warning!!!")
    End Try

End Sub
```

Gambar 17.Program *Button* Tanbah 2

Program diatas berfungsi untuk menambahkan data baru *member* 2. Apabila *button* tambah 2 diklik maka aplikasi akan mengirimkan teks "4tambah" ke Arduino Uno dan nantinya pada Arduino akan menjalankan fungsinya,yang mana fungsi dari *button* tersebut untuk menambahkan data baru *member* 2 yang akan disimpan di EEPROM Arduino. Jika ada kesalahan maka akan muncul peringatan "*error di tombol tambah2*".kesalahan yang ada salah satunya yaitu pengguna belum menekan tombol *edit* pada aplikasi.

9. *Button* Hapus 1

Pada pembahasan ini akan membahas tentang program dari *button* hapus 1. Gambar 18 adalah gambar dari program *button* hapus 1.

```
Private Sub hapus_Click(sender As System.Object, e As System.EventArgs) Handles hapus.Click
    Try
        Dim text As String
        text = "4delete"
            SerialPort1.Write(Text)
            MsgBox("Data berhasil dihapus", MsgBoxStyle.Information, "Pesan")
            tx1.Text = ""
            txnama.Text = ""
    Catch ex As Exception
        MsgBox("error di tombol hapus1", MsgBoxStyle.Critical, "Warning!!!")
    End Try

End Sub
```

Gambar 18. Program *Button* Hapus 1

Program diatas merupakan program untuk menghapus penulisan pada *label* nama dan *label id card*. Ketika *button* hapus 1 diklik maka teks pada label *id card* dan label nama akan terhapus. Jika ada *error* pada *button* hapus 1 maka akan muncul peringatan "*error* di tombol hapus1".

10. *Button* Hapus 2

Pada pembahasan ini akan membahas tentang program dari *button* hapus 2. Gambar 19 adalah gambar dari program *button* hapus 2.

```
Private Sub hapus2_Click(sender As System.Object, e As System.EventArgs) Handles hapus2.Click
    Try
        Dim text As String
        text = "2delete"
            SerialPort1.Write(text)
            MsgBox("Data berhasil dihapus", MsgBoxStyle.Information, "Pesan")
            tx2.Text = ""
            Textnama2.Text = ""
    Catch ex As Exception
        MsgBox("error di tombol hapus2", MsgBoxStyle.Critical, "Warning!!!")
    End Try

End Sub
```

Gambar 19.Program *Button* Hapus 2

Program diatas merupakan program untuk menghapus penulisan pada *label* nama dan *label id card*. Ketika *button* hapus 2 diklik maka teks pada label *id card* dan label nama akan terhapus. Jika ada *error* pada *button* hapus 2 maka akan muncul peringatan "*error* di tombol hapus2".

11. *Button Read* 1

Pada pembahasan ini akan membahas tentang program dari *button read* 1. Gambar 20 adalah gambar dari program *button read* 1.

```
Private Sub read_Click(sender As System.Object, e As System.EventArgs) Handles read.Click
        Try
            Dim text, strKartu As String
                text = "4readdt"
                SerialPort1.Write(Text)
                strKartu = SerialPort1.ReadLine
                tx1.Text = strKartu

        Catch ex As Exception
            MsgBox("error di tombol read1", MsgBoxStyle.Critical, "Warning!!!")

        End Try
    End Sub
```

Gambar 20. Program *Button Read* 1

Program diatas digunakan untuk membaca nomor dari *id card*. Cara kerjanya yaitu apabila *button read* 1 diklik maka akan mengirimkan teks "4readdt" ke Arduino Uno dan nantinya pada Arduino akan menjalankan fungsinya,yang mana fungsi dari *button* tersebut untuk membaca nomor dari *id card* untuk *member* 1. Jika ada kesalahan maka akan muncul peringatan "*error di tombol read*", kesalahan yang ada salah satunya yaitu pengguna belum menekan tombol edit pada aplikasi.

12. *Button Read* 2

Pada pembahasan ini akan membahas tentang program dari *button read* 2. Gambar 21 adalah gambar dari program *button read* 2.

```
Private Sub read2_Click(sender As System.Object, e As System.EventArgs) Handles read2.Click

    Try
        Dim text, strKartu As String
            text = "2readdt"
            SerialPort1.Write(Text)
            strKartu = SerialPort1.ReadLine
            tx2.Text = strKartu

    Catch ex As Exception
        MsgBox("error di tombol read2", MsgBoxStyle.Critical, "Warning!!!")
    End Try
End Sub
```

Gambar 21. Program *Button Read* 2

Program diatas digunakan untuk membaca nomor dari *id card*, cara kerjanya yaitu apabila *button read* 2 diklik maka akan mengirimkan teks "2readdt" ke Arduino Uno dan nantinya pada Arduino akan menjalankan fungsinya, yang mana fungsi dari *button* tersebut untuk membaca nomor dari *id card* untuk *member* 2. Jika ada kesalahan maka akan muncul peringatan "*error di tombol read*2", kesalahan yang ada salah satunya yaitu pengguna belum menekan tombol *edit* pada aplikasi.

E. Alur Instruksi Kerja Lapangan

Pertama saat oli menyentuh sensor *oil level low* maka alarm oli akan berbunyi kemudian *leader curing* produksi membuat *Engineering Job Order* (EJO), selanjutnya tim *engineering* mengecek *level* oli pada manual indikator HPU dan sensor, apabila indikator menunjukkan angka dibawah 60, maka tim *engineering* melakukan perbaikan kebocoran oli. Setelah melakukan perbaikan kebocoran maka tim *engineering* melakukan pengisian oli sampai indikator angka pada tanki HPU menunjukkan angka 65, kemudian melakukan *reset alarm* pada panel HPU (dengan menekan tombol *alarm reset*). Langkah berikutnya yaitu *leader* produksi melakukan pengecekan *alarm* serta memastikan bahwa *engineering* sudah melakukan pengisian oli. Jika sudah *leader* produksi melakukan *tag id card* pada RFID *reader* untuk melakukan *reset alarm* oli.

Bedasarkan pengambilan data waktu *reset alarm Oil Level Low* pada tabel diatas maka dapat disimpulkan bahwa penggunaan RFID dapat mempersingkat waktu dalam me-*reset* alarm *oil low*. Rata-rata waktu yang digunakan untuk me-*reset alarm oil low* yaitu sebelum modifikasi waktu yang digunakan rata-rata 09,04 detik dan untuk sesudah modifikasi yaitu 02,71 detik.

IV. KESIMPULAN

Kesimpulan yang dapat diambil pada Tugas Akhir ini untuk meningkatkan sistem pada sistem *reset alarm Oil Level Low Hydraulic Pump Unit* (HPU) lebih bersifat *high security* dan pembuatan aplikasi untuk mempermudah penambahan data *member* baru adalah sebagai berikut:

1. Modifikasi *reset alarm oil level low* dapat membuat sistem lebih bersifat *high security* dan lebih mudah saat penggunaanya sesuai dengan rancangan, alat ini terdiri dari beberapa komponen yaitu RFID, Arduino Uno, *Buzzer*, PLC Mitsubishi, Kabel, Akrilik, Relay KY-09, *Fan* DC 12V, *mini breadboard* dan adaptor.
2. Aplikasi penambahan data *member* baru dibuat menggunakan Visual Studio, dan telah penulis kelompokan macam-macam pemrogramannya pada Bab 4, yang terdiri dari program *Button* Konek, program *Button Scan*, Program *Form*, Program *Button* tidak Konek, Program *Button Edit*, Program *Button Mode* Normal, Program *Button* Tambah 1 dan 2, Program *Button* Hapus 1 dan , Program *Button Read* 1 dan 2. Hasil uji coba penulis dari 5 kali percobaan transfer data penambahan *member* baru 100% berhasil.

DAFTAR PUSTAKA

[1] Hardi, "Perancangan Sistem Kontrol Pengisian Oli Hidrolik Pada Mesin Curing Tire Line G-H Plant K Berbasis Programmable Logic Controller dan Human Machine Interface," 2016.

[2] M. Yusup, P. Anggriawan, P. Mantovani, and R. Saputro, "Modifikasi Alat Pengingat Waktu Shalat Bebasis Arduino di Mushola," vol. 3, no. 1802048, 2020.

[3] A. A. Marzuki, "Rancang Bangun SistemSmart Home Berbasis Mikrokontroler Arduino Uno," 2017.

[4] S. Setyani, "Rancang Bangun Alat Pengaman Brankas Menggunakan RFID (Radio Frequency Identification) Dengan Memanfaatkan E-KTP Sebagai Tag Berbasis Arduino," 2016.

[5] A. Mubyarto, W. Hp, A. Taryana, and M. Munawar, "Perancangan Prototipe Sistem Konveyor di Industri Dilengkapi Dengan Sistem Pemisah Benda Berdasarkan Warna,Ukuran dan Jenis Benda Berbasis PLC Mitsubishi FX2N," vol. 18, no. 1, pp. 7–14, 2017.

[6] A. R. Prananda, “Pengembangan Sistem Informasi Kesiswaan Menggunakan Framework VB.Net di Pusat Pendidikan dan Pelatihan Kerja KBRI Singapura,” 2017.