

SIMULASI SISTEM *SMART PARKING*

PROYEK AKHIR

Laporan akhir ini dibuat dan diajukan untuk memenuhi salah satu syarat kelulusan Diploma III Politeknik Manufaktur Negeri Bangka Belitung



Disusun Oleh :

Indah Fitria Adila NIRM : 0031716

Yuka Mandiri NIRM : 0031730

**POLITEKNIK MANUFAKTUR NEGERI
BANGKA BELITUNG
TAHUN 2020**

LEMBAR PENGESAHAN

**JUDUL PROYEK AKHIR
SIMULASI SISTEM *SMART PARKING***

Oleh :

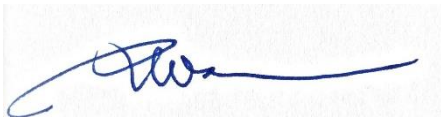
Indah Fitria Adila NIRM : 0031716

Yuka Mandiri NIRM : 0031730

Laporan akhir ini telah disetujui dan disahkan sebagai salah satu syarat kelulusan
Program Diploma III Politeknik Manufaktur Negeri Bangka Belitung

Menyetujui,

Pembimbing 1



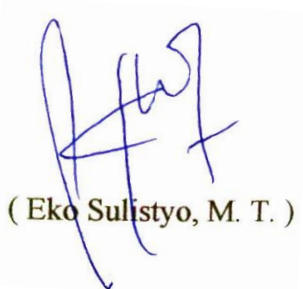
(Irwan, M.Sc., Ph.D.)

Pembimbing 2



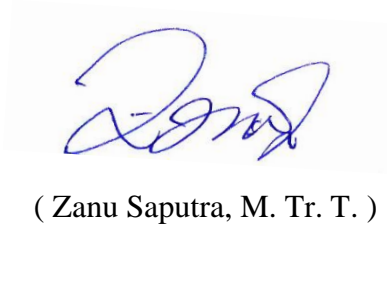
(Charlotha, M.Tr.T.)

Penguji 1



(Eko Sulisty, M. T.)

Penguji 2



(Zanu Saputra, M. Tr. T.)

Penguji 3



(Yudhi, M. T.)

PERNYATAAN BUKAN PLAGIAT

Yang bertanda tangan dibawah ini :

Nama Mahasiswa 1 : Indah Fitria Adila NIRM : 0031716

Nama Mahasiswa 2 : Yuka Mandiri NIRM : 0031730

Dengan Judul : *SIMULASI SISTEM SMART PARKING*

Menyatakan bahwa laporan akhir ini adalah hasil kerja kami sendiri dan bukan merupakan plagiat. Pernyataan ini kami buat dengan sebenarnya dan bila ternyata dikemudian hari ternyata melanggar pernyataan ini, kami bersedia menerima sanksi yang berlaku.

Sungailiat, 27 Agustus 2020

Nama Mahasiswa

Tanda Tangan

1. Indah Fitria Adila



2. Yuka Mandiri



ABSTRAK

Meningkatnya aktivitas mobile, orang dan kendaraan menjadi salah satu dampak akibat majunya teknologi, salah satunya adalah meningkatnya penggunaan kendaraan pribadi yang tidak diseimbangkan dengan area parkir dan sistem pelayanan parkir yang efisien dan memadai. Sistem pelayanan parkir di Indonesia telah menerapkan sistem otomatisasi, namun itu dirasa masih ada kekurangan karena masih sulit bagi pengendara untuk sekedar mencari tempat parkir yang kosong dan membutuhkan waktu yang cukup lama. Untuk mengatasi masalah tersebut maka diperlukan sistem pelayanan parkir yang membantu mengarahkan dan memberikan informasi kepada pengendara mengenai slot tempat parkir yang kosong. Perancangan sistem parkir yang akan dibuat bertujuan agar pengendara mobil yang hendak masuk area parkir mengetahui jumlah area parkir yang tersedia melalui LCD(Liquid Crystal Display) pada pintu masuk area parkir. Pada jalan masuk dan keluar area parkir terdapat palang parkir yang menggunakan servo yang akan terbuka jika sensor ultrasonik mendeteksi adanya kendaraan. Rancangan ini menggunakan Arduino Mega 2560 sebagai Mikrokontroler. Untuk tampilan pada monitor operator menggunakan Visual Studio dan pengolahan database menggunakan Microsoft Access. Dari hasil pengujian yang telah dilakukan Arduino Mega 2560 dapat beroperasi sesuai dengan rangkai dan perintah yang di instruksikan. Sensor ultrasonik bisa mendeteksi tetapi posisi benda harus pas pada sensor, penggunaan database bisa dilakukan dan sistem pembayaran dilakukan secara manual oleh operator. Jadi dapat disimpulkan proyek akhir kami bisa melakukan fungsinya dengan baik tetapi memerlukan operator untuk pengambilan karcis dan pembayaran.

Kata Kunci : *Sistem Smart Parking, Arduino Mega 2560, Sensor Ultrasonik, Visual Studio, Database*

ABSTRACT

The increasing activity of mobile, people and vehicles is one of the impacts due to advancing technology, one of which is the increasing use of private vehicles that are not balanced with parking areas and an efficient and adequate parking service system. The parking service system in Indonesia has implemented an automation system, but it is felt that there are still shortcomings because it is still difficult for motorists to simply find an empty parking space and it takes a long time. To overcome this problem, a parking service system is needed that helps direct and provide information to drivers about empty parking slots. The parking system design that will be made aims to make car drivers who want to enter the parking area know the number of parking areas available through the LCD (Liquid Crystal Display) at the entrance to the parking area. At the entrance and exit of the parking area, there is a parking bar that uses a servo that will open if the ultrasonic sensor detects a vehicle. This design uses the Arduino Mega 2560 as a microcontroller. For display on the operator monitor using Visual Studio and database management using Microsoft Access. From the results of the tests that have been carried out, the Arduino Mega 2560 can operate in accordance with the series and commands instructed. The ultrasonic sensor can detect but the position of the object must fit on the sensor, database use can be done and the payment system is done manually by the operator. So it can be concluded that our final project can do its job well but requires an operator for ticket collection and payment.

Keyword : *Sistem Smart Parking, Arduino Mega 2560, Sensor Ultrasonik, Visual Studio, Database*

KATA PENGANTAR

Puji dan syukur penulis panjatkan kepada Tuhan Yang Maha Esa karena berkat rahmat dan karunia-Nya penulis dapat menyelesaikan laporan Proyek Akhir dengan judul “SIMULASI SISTEM *SMART PARKING*” ini dengan baik dan tepat waktu. Laporan ini disusun sebagai sebagai salah satu persyaratan mahasiswa dalam menyelesaikan Proyek Akhir ini.

Dalam Proyek Akhir ini alat yang penulis kerjakan bertujuan untuk membantu pemilik kendaraan untuk menuju tempat parkir yang tersedia serta membantu petugas dalam melakukan *monitoring* terhadap lahan parkir, dengan menggunakan sistem kontrol dari Mikrokontroler yaitu Arduino Mega 2560 yang dihubungkan dengan LCD dan komputer. Dalam pembuatan Proyek Akhir ini penulis mengakui bahwa selesainya Proyek Akhir ini tidak lepas dari bantuan dan dukungan dari pihak-pihak tertentu. Oleh karena itu penulis mengucapkan terima kasih kepada :

1. Bapak Irwan, Ph.D, selaku pembimbing 1 dalam Proyek Akhir ini
2. Ibu Charlotha, M.Tr.T, selaku pembimbing 2 dalam Proyek Akhir ini.
3. Bapak Eko Sulisty, M.T, selaku Kepala Prodi Teknik Elektronika dan Informatika Politeknik Manufaktur Negeri Bangka Belitung.
4. Bapak Aan Febriansyah, M.T, selaku Kepala Jurusan Teknik Elektronika dan Informatika Politeknik Manufaktur Negeri Bangka Belitung sekaligus wali mahasiswa.
5. Bapak I Made Andik Setiawan, M.Eng., Ph.D., selaku Direktur Politeknik Manufaktur Bangka Belitung.
6. Seluruh staf pengajar dan karyawan di Politeknik Manufaktur Negeri Bangka Belitung.
7. Orang tua beserta keluarga yang lainnya, yang selalu memberikan kasih sayang, doa serta dukungan kepada penulis.
8. Teman-teman yang telah ikut mendukung dan memberikan bantuan serta masukan dalam pembuatan Proyek Akhir ini.

9. Pihak-pihak lain yang telah memberikan bantuan secara langsung maupun tidak langsung yang tidak dapat disebutkan satu per satu.

Dalam laporan Proyek Akhir ini penulis menyadari masih terdapat kekurangan, oleh karena itu penulis sangat mengharapkan saran, kritik dan masukan yang bersifat membangun dalam rangka perbaikan laporan ini. Demikianlah laporan ini dibuat dan semoga laporan ini dapat bermanfaat dan menambah wawasan bagi para pembaca. Akhir kata penulis mengucapkan terima kasih.

Sungailiat, 3 September 2020

Penulis

DAFTAR ISI

| | |
|---|------|
| LEMBAR PENGESAHAN | ii |
| PERNYATAAN BUKAN PLAGIAT | iii |
| ABSTRAK | iv |
| ABSTRACT | v |
| KATA PENGANTAR | vi |
| DAFTAR ISI | viii |
| DAFTAR TABEL | xi |
| DAFTAR GAMBAR | xii |
| DAFTAR LAMPIRAN | xiv |
| BAB I PENDAHULUAN | 1 |
| 1.1 Latar Belakang | 1 |
| 1.2 Perumusan Masalah | 2 |
| 1.3 Batasan Masalah | 2 |
| 1.4 Tujuan | 2 |
| BAB II DASAR TEORI | 3 |
| 2.1 <i>Smart Parking System</i> | 3 |
| 2.2 Mikrokontroler | 3 |
| 2.1.1 Spesifikasi Arduino Mega 2560 | 4 |
| 2.2 Sensor Ultrasonik HC - SR04 | 5 |
| 2.2.1 Prinsip Kerja Sensor Ultrasonik HCSR-04 | 5 |
| 2.3 LCD 16x2 I2C | 6 |
| 2.4 Servo Tower Pro SG90 | 7 |
| 2.5 Arduino IDE | 8 |
| 2.6 Visual Studio 2012 (VB.Net) | 9 |
| 2.7 Basis Data | 9 |
| 2.8 Microsoft Access 2010 | 10 |
| BAB III METODE PELAKSANAAN | 11 |
| 3.1 Pengumpulan Data dan Informasi | 11 |

| | | |
|------------------------|---|----|
| 3.2 | Desain Rancangan <i>Hardware</i> dan <i>Software</i> | 12 |
| 3.4 | Uji coba alat | 13 |
| 3.5 | Analisis Data | 14 |
| BAB IV PEMBAHASAN..... | | 15 |
| 4.1 | Perancangan Sistem Kontrol..... | 15 |
| 4.2 | Perancangan dan Pembuatan <i>Hardware</i> | 18 |
| 4.2.1 | Perancangan <i>Hardware</i> Secara Mekanik | 19 |
| 4.2.2 | Pembuatan <i>Hardware</i> Secara Mekanik..... | 20 |
| 4.2.3 | Perancangan dan Pembuatan <i>Hardware</i> Secara Elektrik..... | 21 |
| 4.2.3.1 | Perancangan Pembuatan <i>Hardware</i> Sensor Ultrasonik | 22 |
| 4.2.3.2 | Perancangan Pembuatan <i>Hardware</i> LCD 16x2 I2C | 23 |
| 4.2.3.3 | Perancangan Pembuatan <i>Hardware</i> Servo Tower SG90 | 24 |
| 4.3 | Pengujian <i>Hardware</i> | 25 |
| 4.3.1 | Pengujian Sensor Ultrasonik | 25 |
| 4.3.2 | Pengujian LCD 16x2 I2C..... | 28 |
| 4.3.3 | Pengujian Servo Tower Pro SG90 dengan Sensor Ultrasonik | 28 |
| 4.3.4 | Pengujian sensor Ultrasonik dengan LCD 16x2 I2C | 29 |
| 4.4 | Perancangan dan Pembuatan <i>Software</i> | 30 |
| 4.4.1 | Perancangan dan Pembuatan <i>Database</i> | 30 |
| 4.4.2 | Perancangan dan Pembuatan <i>Interface</i> Microsoft Visual Studio..... | 31 |
| 4.4.2.1 | Komunikasi Serial pada Microsoft Visual Studio 2012..... | 36 |
| 4.4.2.1.1 | Uji Coba Koneksi Arduino dengan Visual Studio | 37 |
| 4.4.2.2 | Pemisahan Data Sensor di Visual Studio | 38 |
| 4.4.2.2.1 | Uji Coba Pemisahan Data | 40 |
| 4.4.2.3 | Koneksi Visual Studio dengan <i>Database</i> | 41 |
| 4.4.2.3.1 | Uji Coba Koneksi dengan <i>Database</i> | 43 |
| 4.4.2.4 | Koneksi Visual Studio dengan <i>Printer</i> | 45 |
| 4.4.2.4.1 | Uji Coba <i>Print</i> Karcis..... | 45 |
| 4.5 | Standar Operasional Prosedur (SOP) Proses Kerja Sistem Parkir | 46 |
| BAB V PENUTUP..... | | 47 |
| 5.1 | Kesimpulan | 47 |

| | |
|----------------------|----|
| 5.2 Saran..... | 47 |
| DAFTAR PUSTAKA | 48 |

DAFTAR TABEL

| | |
|--|----|
| Tabel 2. 1 Spesifikasi Arduino Mega 2560..... | 4 |
| Tabel 4. 1 Penempatan pin komponen pada Arduino | 16 |
| Tabel 4. 2 Pengujian jarak sensor ultrasonik | 26 |
| Tabel 4. 3 Pengujian data sensor..... | 27 |
| Tabel 4. 4 Pengujian sensor ultrasonik dengan LCD..... | 29 |

DAFTAR GAMBAR

| | |
|--|----|
| Gambar 2. 1 Mikrokontroler Arduino Mega 2560..... | 4 |
| Gambar 2. 2 Sensor ultrasonik HC-SR04 | 5 |
| Gambar 2. 3 Cara kerja sensor ultrasonik HCSR-04 | 6 |
| Gambar 2. 4 LCD 16x2..... | 6 |
| Gambar 2. 5 <i>Module</i> I2C..... | 7 |
| Gambar 2. 6 Servo Tower Pro SG90 | 8 |
| Gambar 2. 7 Tampilan Arduino IDE..... | 8 |
| Gambar 2. 8 Tampilan jendela Visual Studio 2012 | 9 |
| Gambar 2. 9 Tampilan Ms.Access | 10 |
| Gambar 3. 1 Blok diagram metode pelaksanaan..... | 11 |
| Gambar 3. 2 Blok diagram sistem <i>smart parking</i> | 12 |
| Gambar 4. 1 Blok diagram sistem kontrol | 15 |
| Gambar 4. 2 <i>Flowchart</i> cara kerja simulasi sistem smart parking..... | 18 |
| Gambar 4. 3 Desain base sistem smart parking | 19 |
| Gambar 4. 4 Hasil akhir pembuatan <i>base</i> parkir bagian atas..... | 20 |
| Gambar 4. 5 Skematik simulasi sistem <i>smart parking</i> | 21 |
| Gambar 4. 6 Hasil akhir <i>wiring</i> pada komponen | 22 |
| Gambar 4. 7 Skematik sensor ultrasonik..... | 22 |
| Gambar 4. 8 Pemograman sensor ultrasonik..... | 23 |
| Gambar 4. 9 Skematik LCD 16x2 I2C..... | 23 |
| Gambar 4. 10 Pemograman LCD 16x2 I2C..... | 24 |
| Gambar 4. 11 Skematik Servo Tower Pro SG90 | 24 |
| Gambar 4. 12 Pemograman Servo Tower Pro SG90 | 25 |
| Gambar 4. 13 Hasil pembacaan jarak pada sensor..... | 26 |
| Gambar 4. 14 Pengujian baca data sensor ultrasonik..... | 27 |
| Gambar 4. 15 Hasil pengujian LCD 16x2 I2C..... | 28 |
| Gambar 4. 16 Hasil pengujian Servo Tower Pro SG90 | 29 |
| Gambar 4. 17 Pengujian sensor ultrasonik dengan LCD | 30 |

| | |
|---|----|
| Gambar 4. 18 Keadaan LCD jika parkir penuh..... | 30 |
| Gambar 4. 19 <i>Database</i> Ms.Access | 31 |
| Gambar 4. 20 <i>Tab</i> menu utama..... | 32 |
| Gambar 4. 21 Menu utama dan <i>login</i> | 32 |
| Gambar 4. 22 Akses menu utama | 33 |
| Gambar 4. 23 Menu <i>monitoring</i> | 34 |
| Gambar 4. 24 Menu <i>print</i> | 35 |
| Gambar 4. 25 Menu pembayaran parkir | 36 |
| Gambar 4. 26 Pemograman komunikasi Arduino dengan Visual Studio | 37 |
| Gambar 4. 27 Koneksi Arduino dengan Visual Studio..... | 37 |
| Gambar 4. 28 <i>Textbox</i> pemisahan data | 38 |
| Gambar 4. 29 Inisialisasi variabel..... | 38 |
| Gambar 4. 30 Pemograman pemisah data..... | 39 |
| Gambar 4. 31 Pemograman <i>input</i> data Arduino..... | 40 |
| Gambar 4. 32 Pemisahan data ke dalam <i>textbox</i> | 41 |
| Gambar 4. 33 Perubahan indikator warna pada <i>picturebox</i> | 41 |
| Gambar 4. 34 Pembuatan <i>module</i> koneksi..... | 42 |
| Gambar 4. 35 Menampilkan <i>database</i> ke dalam <i>datagridview</i> | 42 |
| Gambar 4. 36 Simpan data ke dalam <i>database</i> | 43 |
| Gambar 4. 37 Tampilan <i>database</i> di Visual Studio | 44 |
| Gambar 4. 38 Tampilan biaya parkir | 44 |
| Gambar 4. 39 Pemograman koneksi <i>printer</i> dengan Visual Studio..... | 45 |
| Gambar 4. 40 Hasil <i>print</i> karcis | 45 |
| Gambar 4. 41 SOP proses kerja sistem parkir..... | 46 |

DAFTAR LAMPIRAN

1. Lampiran : Daftar Riwayat Hidup
2. Lampiran : Program Arduino Mega 2560
3. Lampiran : Program Visual Studio
4. Lampiran : Petunjuk Penggunaan Aplikasi Bagi Operator
5. Lampiran : Datasheet

BAB I

PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi saat ini berpengaruh pada kebutuhan hidup masyarakat. Salah satunya adalah meningkatnya aktivitas *mobile*, orang maupun kendaraan. Saat ini banyak masyarakat yang menggunakan kendaraan pribadi namun tidak diseimbangkan dengan *area* parkir yang memadai. Seperti yang diketahui *area* parkir masih mengandalkan petugas parkir untuk mengatur kendaraan masuk maupun keluar dan disini terlihat bahwa kurangnya petugas parkir dalam memperhatikan kapasitas *area* parkir, sehingga membawa dampak kepada pengendara. Dalam hal ini pengendara membutuhkan waktu untuk berkeliling terlebih dahulu mencari slot parkir yang tersedia. Sebuah survei yang dikutip dari Viva.co.id menyatakan bahwa butuh 21 menit untuk mencari tempat parkir di Jakarta [1].

Di Indonesia sistem pelayanan parkir sudah menggunakan otomatisasi dalam pengoperasiannya, namun masih belum efektif karena masih kurangnya informasi yang diberikan kepada pengendara mengenai lokasi parkir. Hal ini menyebabkan pengendara harus berkeliling terlebih dahulu untuk sekedar mencari slot tempat parkir yang kosong dan itu membutuhkan waktu cukup lama dan akan merugikan pengendara.

Berdasarkan hal tersebut tentunya dapat diatasi dengan meningkatkan sistem pelayanan parkir yang lebih *modern* dan menguntungkan bagi pengguna parkir maupun perusahaan besar atau tempat umum lainnya. Maka dari itu diperlukan sebuah kendali parkir cerdas (*smart parking*) yang dapat memudahkan pengelola dalam melakukan pemantauan terhadap slot tempat parkir serta pengguna dalam menemukan tempat parkir kosong.

Pada proyek akhir yang telah dibuat oleh Galih Raditya Pradana dari Universitas Negeri Yogyakarta pada tahun 2015 yang berjudul “ *Smart Parking* Berbasis Arduino Uno” [2] dikembangkan sistem parkir cerdas yang

menginformasikan jumlah slot tempat parkir yang tersedia, keterangan lokasi tempat parkir yang kosong serta slot tempat parkir terdekat melalui *display* monitor. Walau demikian masih ada kekurangan pada proyek tersebut dikarenakan masih terfokus dalam memberikan informasi mengenai slot dan jumlah tempat parkir yang tersedia dan tidak mengarahkan objek yaitu pengendara langsung ke tempat parkir yang kosong. Maka dalam proyek kali ini dikembangkan Simulasi Sistem *Smart Parking* yang dapat membantu mengarahkan dan memberikan informasi kepada pengendara mengenai status maupun lokasi tempat parkir yang kosong.

1.2 Perumusan Masalah

Berdasarkan pemaparan dalam latar belakang dapat dirumuskan permasalahan yaitu bagaimana membuat simulasi sistem *smart parking* yang dapat memberikan informasi sekaligus mengarahkan pengendara ke slot tempat parkir.

1.3 Batasan Masalah

Dalam pembuatan proyek akhir simulasi sistem *smart parking* ini terdapat batasan masalah yaitu sebagai berikut:

1. Sensor ultrasonik yang digunakan untuk mendeteksi kendaraan diaplikasikan pada setiap slot tempat parkir.
2. Penyimpanan data slot masuk parkir dan nomor karcis dikendalikan oleh operator.
3. Pembayaran parkir dan *print* karcis dikendalikan oleh operator.

1.4 Tujuan

1. Membantu pemilik kendaraan untuk menuju tempat parkir yang tersedia.
2. Membantu petugas dalam *monitoring* slot tempat parkir yang kosong dan tersedia.

BAB II

DASAR TEORI

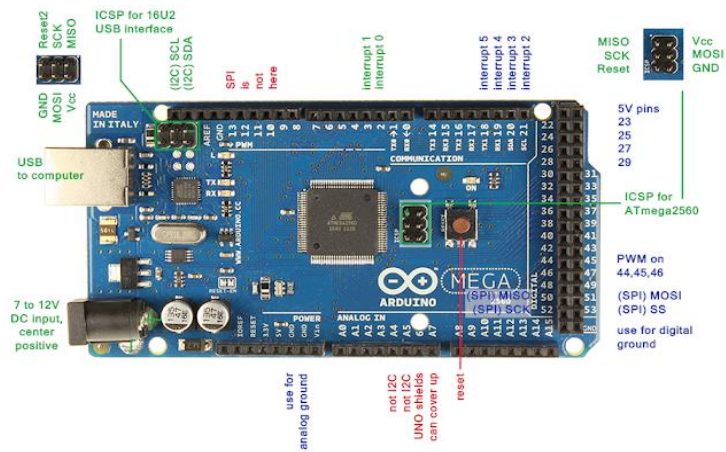
2.1 Smart Parking System

Teknologi dengan mengandalkan otomatisasi telah banyak digunakan salah satunya adalah sistem parkir. Sistem parkir pintar adalah salah satu teknologi yang menggunakan sistem otomatisasi dalam penerapannya, dimana berfokus pada *monitoring* dan keamanan akses. Dengan menggunakan sensor pada sistem dan dikontrol oleh perangkat komputer serta juga adanya komunikasi dengan jaringan maka *smart parking system* bisa menjadikan tata kelola parkir menjadi lebih teratur [3].

2.2 Mikrokontroler

Mikrokontroler adalah sebuah *chip* yang berfungsi sebagai pengontrol rangkaian elektronik dan terdapat program di dalamnya. Mikrokontroler pada umumnya terdiri dari *processor*, memori, *input/output* serta unit pendukung seperti *Analog-to-Digital Converter* (ADC) yang sudah terintegrasi di dalamnya. Dengan demikian Mikrokontroler adalah sebuah alat elektronika digital yang dapat dikontrol dengan program yang dapat ditulis maupun di edit sesuai dengan yang diinginkan dan mempunyai masukan dan keluaran [4]. Dari sekian banyaknya jenis Mikrokontroler yang paling sering digunakan adalah Arduino. Dalam hal ini Arduino yang digunakan dalam pembuatan simulasi sistem *smart parking* adalah Arduino Mega 2560.

Arduino Mega 2560 merupakan sebuah *board* Arduino yang menggunakan IC Mikrokontroler ATmega 2560. Arduino ini memiliki pin I/O yang relatif banyak yaitu 54 digital *input/output* yang 15 buah di antaranya dapat digunakan sebagai *output* PWM, selain itu Arduino ini juga memiliki 16 buah analog *input*, 4 UART (*port serial* perangkat keras), dilengkapi dengan kristal 16. Mhz, *power* dari USB ke PC/Laptop atau melalui *jack* DC pakai adaptor 7-12 V DC serta tombol *reset* [5].



Gambar 2. 1 Mikrokontroler Arduino Mega 2560 [6]

2.1.1 Spesifikasi Arduino Mega 2560

Tabel 2. 1 Spesifikasi Arduino Mega 2560 [6]

| Komponen | Spesifikasi |
|-------------------------------------|---|
| Mikrokontroler | Atmega 2560 |
| Tegangan Operasional | 5 V |
| Tegangan <i>Input</i> (rekomendasi) | 7-12V |
| Tegangan <i>Input</i> (limit) | 6-20V |
| Pin Digital I/O | 54 (of which 15 provide PWM output) |
| Pin Analog <i>Input</i> | 16 |
| Arus DC per Pin I/O | 20 mA |
| Arus DC untuk Pin 3.3 V | 50 mA |
| Memori Flash | 256 KB of which 8 KB used by bootloader |
| SRAM | 8 KB |
| EEPROM | 4 KB |
| Clock Speed | 16 MHz |
| LED_BUILTIN | 13 |
| Panjang | 101.52 mm |
| Lebar | 53.3 mm |
| Berat | 37 g |

2.2 Sensor Ultrasonik HC - SR04

Sensor ultrasonik merupakan sebuah sensor pendeteksi jarak yang bekerja berdasarkan gelombang ultrasonik (bunyi ultrasonik) yang dipancarkan terhadap benda yang kemudian dipantulkan kembali. Sensor ultrasonik HC-SR04 ini adalah sensor yang sudah siap pakai dimana dalam rangkaiannya telah terpasang *transducer* ultrasonik yang berfungsi sebagai *transmitter* yaitu pemancar gelombang suara ultrasonik dan *receiver* sebagai penerima ultrasonik [7].

Sensor ini memiliki 4 pin yang terdiri dari pin Vcc, Gnd, *Trigger* dan *Echo*. Pin Vcc pada sensor untuk listrik positif dan Gnd untuk *ground*-nya, untuk pin *Trigger* digunakan sebagai keluaran sinyal dari sensor atau pemancar sinyal dan pin *Echo* untuk menangkap sinyal pantul dari benda [7].



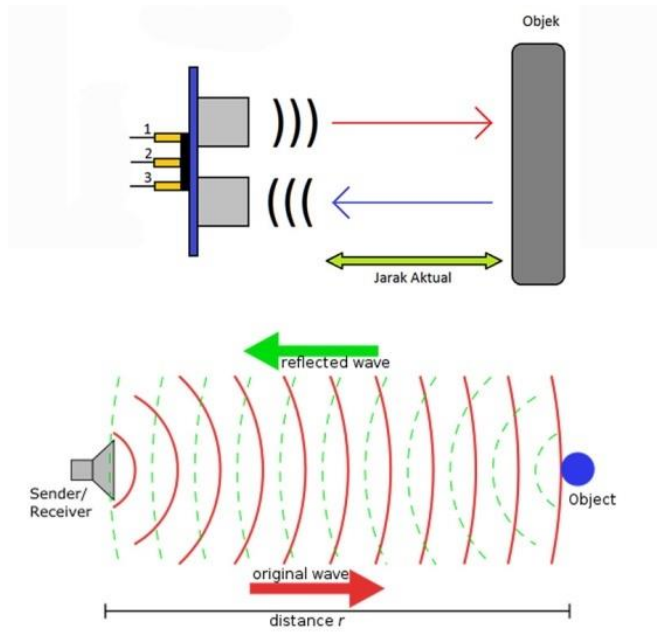
Gambar 2. 2 Sensor ultrasonik HC-SR04 [8]

2.2.1 Prinsip Kerja Sensor Ultrasonik HCSR-04

Berikut prinsip kerja sensor ultrasonik :

- Sinyal dipancarkan oleh pemancar ultrasonik (*transmitter*), dengan frekuensi diatas 40 KHZ Untuk mengukur jarak benda dan durasi waktu tertentu [8].
- Sinyal yang dipancarkan merambat sebagai gelombang bunyi dengan kecepatan sekitar 340 m/s. Ketika terpancarkan suatu benda, maka sinyal tersebut akan dipantulkan oleh benda tersebut [8].
- Setelah gelombang pantulan sampai di *receiver*, maka sinyal akan diproses untuk menghitung jarak benda tersebut. Jarak benda dihitung berdasarkan rumus [8]:

$$\text{Jarak (cm)} = \text{Waktu } (\mu\text{S}) * 0.034 / 2 \dots\dots\dots (2.1)$$



Gambar 2. 3 Cara kerja sensor ultrasonik HCSR-04 [8]

2.3 LCD 16x2 I2C

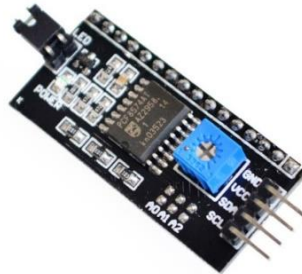
LCD atau *Liquid Crystal Display* adalah suatu jenis media *display* (tampilan) yang dapat menghasilkan data dalam bentuk karakter, huruf, ataupun angka sesuai yang diinginkan dengan melakukan pemrograman. LCD 16 x 2 ini memiliki 2 baris dan 16 kolom [5]. Berikut bentuk fisik dari LCD 16x2 :



Gambar 2. 4 LCD 16x2 [9]

Pada LCD 16x2 menggunakan komponen tambahan yaitu *Inter Integrated Circuit* atau *I2C module*, dimana dengan menggunakan I2C maka LCD akan dikendalikan secara serial dua arah dan akan menghemat penggunaan pin pada

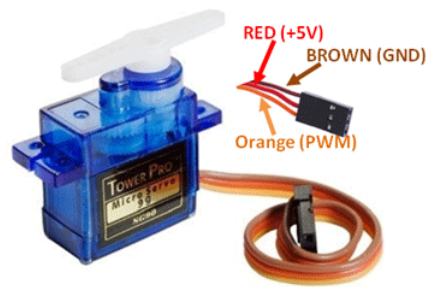
kontroller (Arduino) dibandingkan dengan tidak menggunakan I2C yang akan memakan banyak pin dari kontroller. Sistem I2C terdiri dari saluran SCL (*Serial Clock*) dan SDA (*Serial Data*) yang membawa informasi data antara I2C dengan pengontrolnya [10].



Gambar 2. 5 *Module I2C* [10]

2.4 Servo Tower Pro SG90

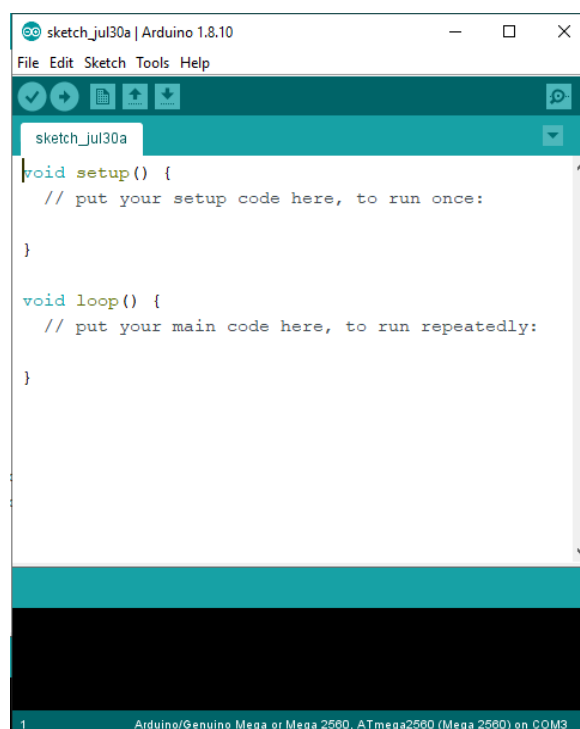
Servo Tower Pro SG90 merupakan jenis servo yang berukuran kecil dan ringan tetapi mempunyai daya yang tinggi. Servo ini bekerja seperti servo pada umumnya namun hanya saja konstruksinya lebih kecil dan dapat berputar 180 derajat (90 derajat setiap arah). Servo ini dapat dikendalikan menggunakan pemrograman dari kontrol Arduino [5]. Jika dipahami dari contoh sinyal PWM yang memiliki frekuensi 50 Hz maka periode PWM harus 20 ms, *on-time* dapat bervariasi dari 1 ms hingga 2 ms. Jadi ketika waktu tepat adalah 1 ms, motor akan berada di 0 ° dan ketika 1,5 ms motor akan menjadi 90 °, demikian pula ketika 2 ms itu akan menjadi 180 °. Jadi dengan memvariasikan waktu tepat dari 1 ms ke 2 ms motor dapat dikontrol dari 0 ° hingga 180 ° [11].



Gambar 2. 6 Servo Tower Pro SG90 [11]

2.5 Arduino IDE

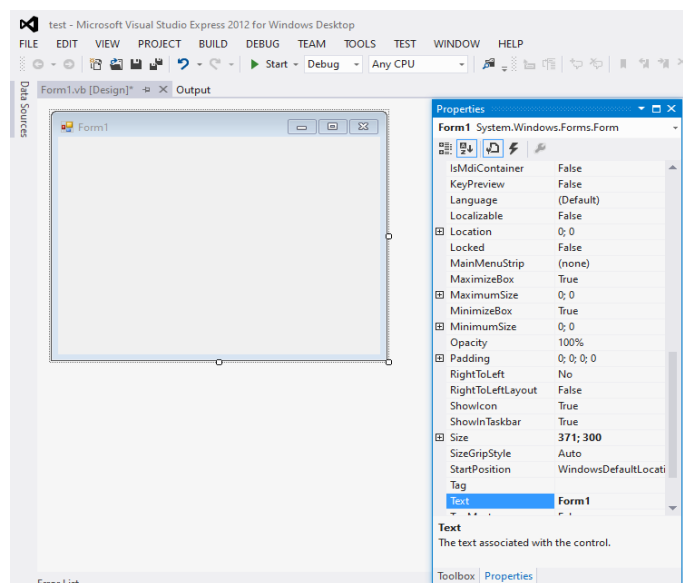
Arduino IDE (*Integrated Development Environment*) adalah *software* yang di gunakan untuk memprogram *board* Arduino, yang di dalamnya merupakan teks editor untuk membuat , mengedit, memvalidasi kode program dan juga meng-*compile* menjadi kode biner yang kemudian meng-*upload* ke dalam memori Mikrokontroler [5]. Program yang dituliskan pada Arduino disebut dengan *sketch* atau *source code* Arduino dengan ekstensi file *source code* .ino. Tampilan jendela dari Arduino IDE dapat dilihat pada Gambar 2. 7.



Gambar 2. 7 Tampilan Arduino IDE

2.6 Visual Studio 2012 (VB.Net)

Visual Studio 2012 atau yang sering disebut dengan VB.Net merupakan sebuah bahasa pemrograman yang menawarkan *Integrated Development Environment* (IDE) visual untuk membuat program perangkat lunak/aplikasi yang berbasis *windows* dengan berbasis GUI (*Grapical User Interface*). Bahasa pemograman yang digunakan dalam Visual Studio cukup sederhana karena menggunakan kata dalam bahasa inggris yang umum digunakan. Di dalam Visual Studio telah menyediakan beberapa pilihan-pilihan untuk menunjang pembuatan apilikasi/ perangkat lunak yang bisa digunakan sesuai dengan kebutuhan. Selain itu Visual Studio juga bisa digunakan dalam lingkungan *net-working* atau apilikasi penggunaan *client-server* [12].



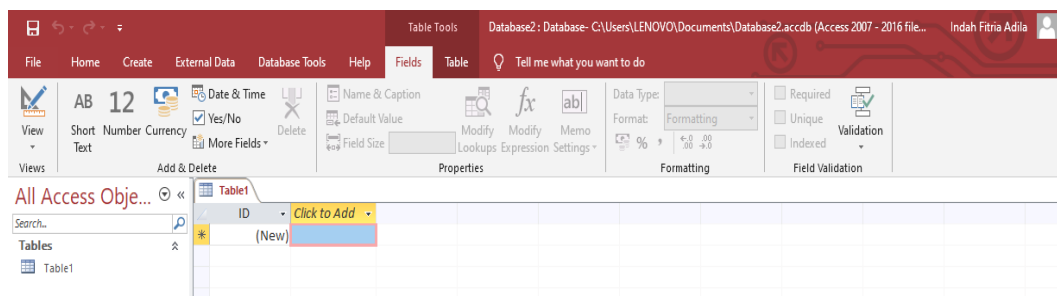
Gambar 2. 8 Tampilan jendela Visual Studio 2012

2.7 Basis Data

Basis data (*database*) adalah kumpulan suatu informasi yang disimpan didalam komputer secara sistematis dimana suatu informasi tersebut dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi.

2.8 Microsoft Access 2010

Microsoft Access adalah sebuah aplikasi sistem manajemen *database* atau (DBMS). Microsoft Access dapat menyimpan berbagai macam informasi yang disebut data dengan mengatur dan mengelolanya sedemikian rupa agar data tersebut mudah dipergunakan kembali. Microsoft Access dari generasi sebelumnya telah menyediakan antar muka dalam bentuk grafis, untuk setiap langkah pembuatan maupun pengelolaan *database* sehingga sangat membantu dalam membangun suatu sistem manajemen *database* [13].



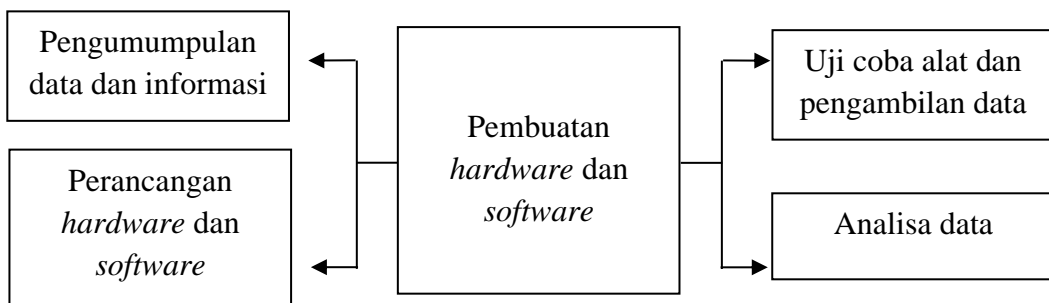
Gambar 2. 9 Tampilan Ms.Access

BAB III METODE PELAKSANAAN

Pada bab ini akan dibahas metode pelaksanaan yang dilakukan dalam proses pembuatan proyek akhir yang berjudul “Simulasi Sistem *Smart Parking* “.

Berikut garis besar tahapan yang dilakukan:

1. Pengumpulan data dan informasi
2. Desain rancangan *hardware* dan *software*
3. Pembuatan *hardware* dan *software*
4. Uji coba alat dan pengambilan data
5. Analisis data



Gambar 3. 1 Blok diagram metode pelaksanaan

Berikut penjelasan dari tahapan - tahapan diatas:

3.1 Pengumpulan Data dan Informasi

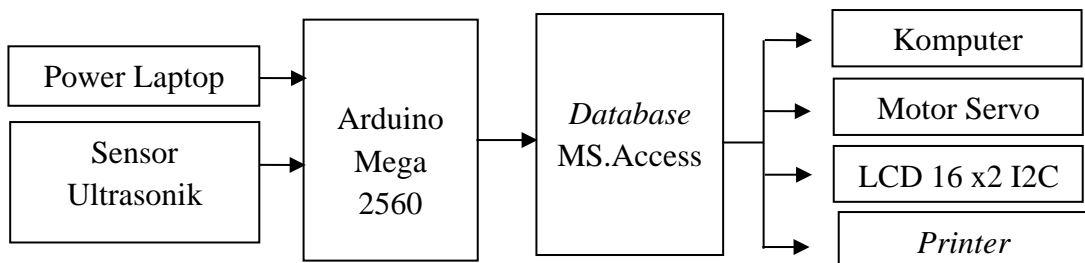
Melakukan pengumpulan data dan informasi tentang sistem *smart parking*. pengumpulan data dan informasi ini bertujuan untuk mengetahui bagaimana sistem itu bekerja, prinsip serta tujuan dari adanya sistem *smart parking*, yang nantinya dapat ditemukan cara untuk mengatasi permasalahan yang terjadi. Informasi dan data yang ditampung akan dijadikan acuan untuk pembuatan simulasi sistem *smart parking* kedepannya.

Selain itu juga mencari informasi mengenai penggunaan komunikasi serial dalam pembuatan *smart parking*, pemrograman *interface* dengan Visual Studio serta cara kerja sensor dan Mikrokontroler.

3.2 Desain Rancangan *Hardware* dan *Software*

Desain *hardware* dan *software* dilakukan untuk mencari bentuk dan sistem dari simulasi *smart parking*. Pada proses perancangan *hardware* meliputi mekanis (konstruksi) dan elektrik. Dalam tahapan perancangan mekanis dilakukan perancangan model alat dengan menggunakan *Corel Draw*.

Sedangkan dalam tahapan secara elektrik yaitu dengan membuat rancangan sistem berdasarkan dari hasil pengumpulan data dan informasi seperti pada blok diagram berikut :



Gambar 3. 2 Blok diagram sistem *smart parking*

Untuk perancangan *software* (*user interface*) yaitu menetapkan fitur yang sesuai untuk digunakan dalam tampilan simulasi, dalam hal ini menggunakan fitur *monitoring* untuk memantau lokasi tempat parkir dan kasir untuk pembayaran parkir.

3.3 Pembuatan *Hardware* dan *Software*

Untuk pembuatan *hardware* berupa pembentukan konstruksi *prototipe*, dengan melakukan penentuan alat dan bahan yang digunakan, serta ukuran yang digunakan.

Berikut tahapan yang dilakukan dalam pembuatan *hardware* secara mekanis:

1. Pembuatan *base* sebagai lahan parkir yang terbuat dari triplek dengan ketebalan 2 cm dan berukuran 51x51 serta *stainless* sebagai pembatas lahan parkir.
2. Pemasangan sensor ultrasonik, Mikrokontroler (Arduino Mega 2560), Servo Tower Pro SG90 serta pemasangan kabel pada *base* parkir.
3. Pembuatan palang parkir dengan menggunakan stik kayu.

Sedangkan untuk pembuatan sistem elektronik dan *software* meliputi:

1. Pemrograman deteksi sensor ultrasonik untuk menghitung slot parkir tersedia yang ditampilkan di dalam LCD 16x2 I2C.
2. Pembuatan *database* parkir.
3. Pemrograman tampilan *monitoring* parkir dan sistem bayar parkir.
4. Pemrograman *print* karcis dan peta *area* parkir.

3.4 Uji coba alat

Setelah proses pembuatan *hardware*, *software* serta sistem maka selanjutnya dilakukan uji coba alat, dimana bertujuan untuk mengetahui apakah kerja alat kita sudah optimal atau sesuai dengan yang diinginkan atau tidak. terdapat dua tahapan dalam uji coba alat:

1. Uji coba *Hardware*
 - Uji coba Arduino dengan sensor ultrasonik
 - Uji coba Arduino dengan Servo dan sensor ultrasonik
 - Uji coba Arduino dengan LCD 16x2 I2C
 - Uji coba Arduino dengan sensor ultrasonik, LCD 16 x 2
2. Uji coba *Software* dan sistem
 - Uji coba koneksi antara Arduino dengan Visual Studio 2012
 - Uji coba antara *database* dengan Visual Studio 2012
 - Uji coba *printer* dengan Visual Studio 2012
3. Uji coba keseluruhan yaitu *hardware* dan *software*

3.5 Analisis Data

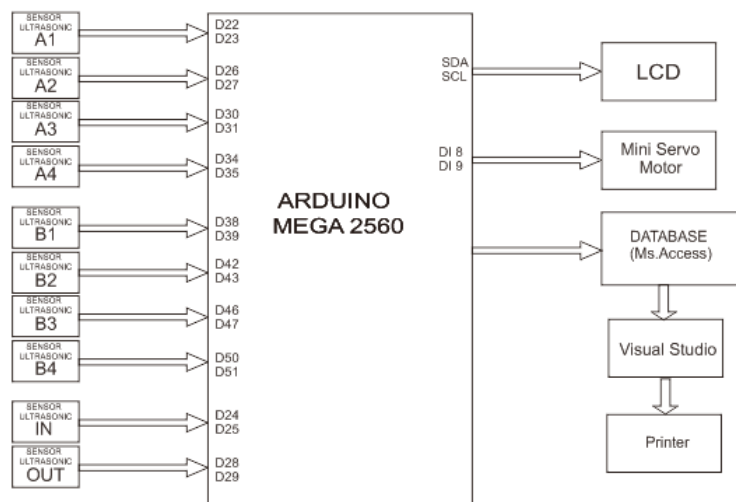
Pada tahap ini dilakukan analisis data berdasarkan hasil uji coba *hardware*, *software* serta sistem apakah secara keseluruhan sesuai dengan yang diinginkan dari segi konstruksi, kontrol serta *interface*.

BAB IV PEMBAHASAN

Pada bab 4 akan membahas proses pembuatan *prototipe* simulasi sistem *smart parking* yang terdiri dari perancangan sistem kontrol, perancangan dan pembuatan *hardware* dan *software*, dan pengujian sistem. Berikut penjelasannya:

4.1 Perancangan Sistem Kontrol

Sistem kontrol yang kami gunakan adalah Mikrokontroler Arduino Mega 2560. Perancangan sistem kontrol akan menentukan hasil yang didapat. Berikut adalah blok diagram sistem kontrol yang akan digunakan untuk sistem parkir berbasis Arduino dan sistem *database*.



Gambar 4. 1 Blok diagram sistem kontrol

Berdasarkan blok diagram Gambar 4. 1 berikut adalah penjelasan pin Arduino Mega 2560 yang dihubungkan ke masing-masing *input* dan *output* pada *prototipe* simulasi sistem *smart parking*.

Tabel 4. 1 Penempatan pin komponen pada Arduino

| No. | Komponen | Pin pada Arduino Mega 2560 |
|-----|-----------------------|--|
| 1 | Sensor Ultrasonik A1 | 22 dan 23 (<i>Triger & Echo</i>) |
| 2 | Sensor Ultrasonik A2 | 26 dan 27 (<i>Triger & Echo</i>) |
| 3 | Sensor Ultrasonik A3 | 30 dan 31 (<i>Triger & Echo</i>) |
| 4 | Sensor Ultrasonik A4 | 34 dan 35 (<i>Triger & Echo</i>) |
| 5 | Sensor Ultrasonik B1 | 38 dan 39 (<i>Triger & Echo</i>) |
| 6 | Sensor Ultrasonik B2 | 42 dan 43 (<i>Triger & Echo</i>) |
| 7 | Sensor Ultrasonik B3 | 46 dan 47 (<i>Triger & Echo</i>) |
| 8 | Sensor Ultrasonik B4 | 50 dan 51 (<i>Triger & Echo</i>) |
| 9 | Sensor Ultrasonik IN | 24 dan 25 (<i>Triger & Echo</i>) |
| 10 | Sensor Ultrasonik OUT | 28 dan 29 (<i>Triger & Echo</i>) |
| 11 | LCD 16x2 I2C | SDA dan SCL |
| 12 | Servo Tower Pro SG90 | 8 dan 9 (Pin PWM) |

Berdasarkan blok diagram di atas, berikut adalah penjelasan dan kegunaan dari alat tersebut :

1. Arduino Mega 2560

Arduino Mega 2560 adalah Mikrokontroler yang di gunakan untuk pembuatan alat proyek akhir dan menggunakan *software* Arduino IDE untuk pemrograman sistem parkir.

2. Sensor Ultrasonik HC-SR04

Sensor ultrasonik merupakan sebuah sensor pendeteksi jarak yang bekerja berdasarkan gelombang ultrasonik (bunyi ultrasonik) yang rangkaiannya telah terpasang *transducer* ultrasonik yang berfungsi sebagai *transmitter dan receiver*. Dalam proyek akhir ini sensor ultrasonik digunakan untuk mendeteksi kendaraan dalam simulasi sistem *smart parking*.

3. LCD 16x2 I2C

LCD 16x2 I2C atau *Liquid Crystal Display* adalah suatu jenis media *display* (tampilan) yang dapat menghasilkan data dalam bentuk karakter,

huruf, ataupun angka yang memiliki 2 baris dan 16 kolom dengan dilengkapi *module I2C* untuk mengurangi penggunaan kabel. LCD 16x2 I2C digunakan untuk menampilkan slot parkir yang tersedia.

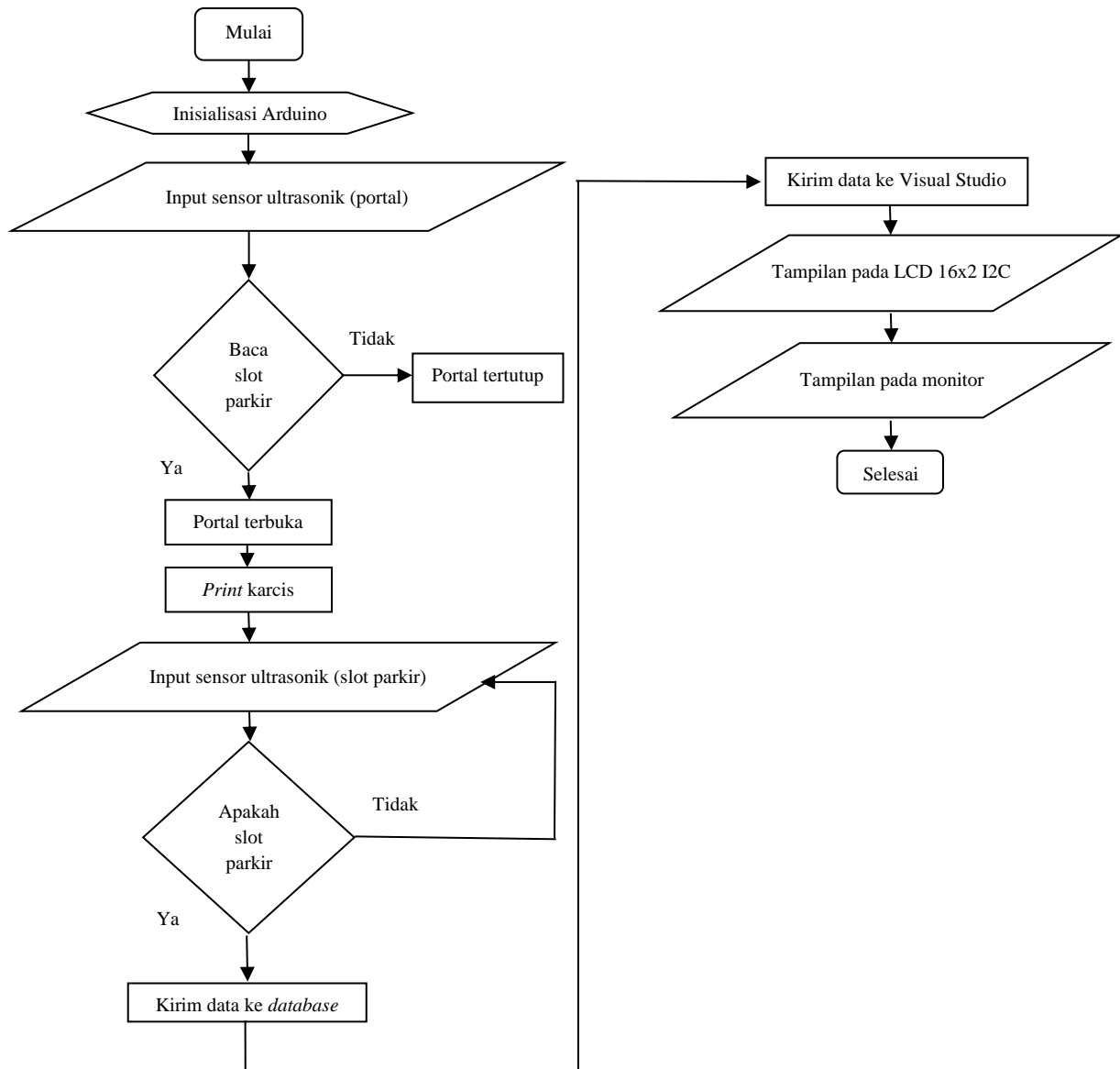
4. Servo Tower Pro SG90

Servo Tower Pro SG90 merupakan jenis servo yang berukuran kecil dan serta mempunyai daya yang tinggi. Servo ini bekerja seperti servo pada umumnya dan berputar 180 derajat (90 derajat setiap arah). Servo ini dapat dikendalikan menggunakan pemrograman dari kontrol Arduino. Servo ini digunakan untuk membuka dan menutup portal pada *area* parkir.

5. Komputer/ Laptop

Komputer/ Laptop ini digunakan untuk menampilkan *software monitoring* slot tempat parkir yang tersedia serta pembayaran parkir yang telah dibuat dalam Visual Studio.

Pada proyek akhir ini menggunakan Arduino mega 2560 sebagai kontroller serta memanfaatkan sensor ultrasonik pada setiap *area* parkir dan juga terdapat satu buah portal pada pintu masuk dan satu lagi pada pintu keluar, dalam hal ini menggunakan sensor ultrasonik untuk membuka portal atau menggerakkan servo jika mendeteksi adanya mobil masuk dan keluar *area* parkir. Jika slot parkir tersedia maka portal akan terbuka dan karcis akan dicetak. Kemudian setelah mobil menempati ke slot yang dituju maka akan mengirimkan data slot parkir ke dalam *database* yang kemudian di tampilkan di Visual Studio, LCD dan monitor. Berikut adalah *flowchart* cara kerja alat sistem parkir pada Gambar 4. 2.



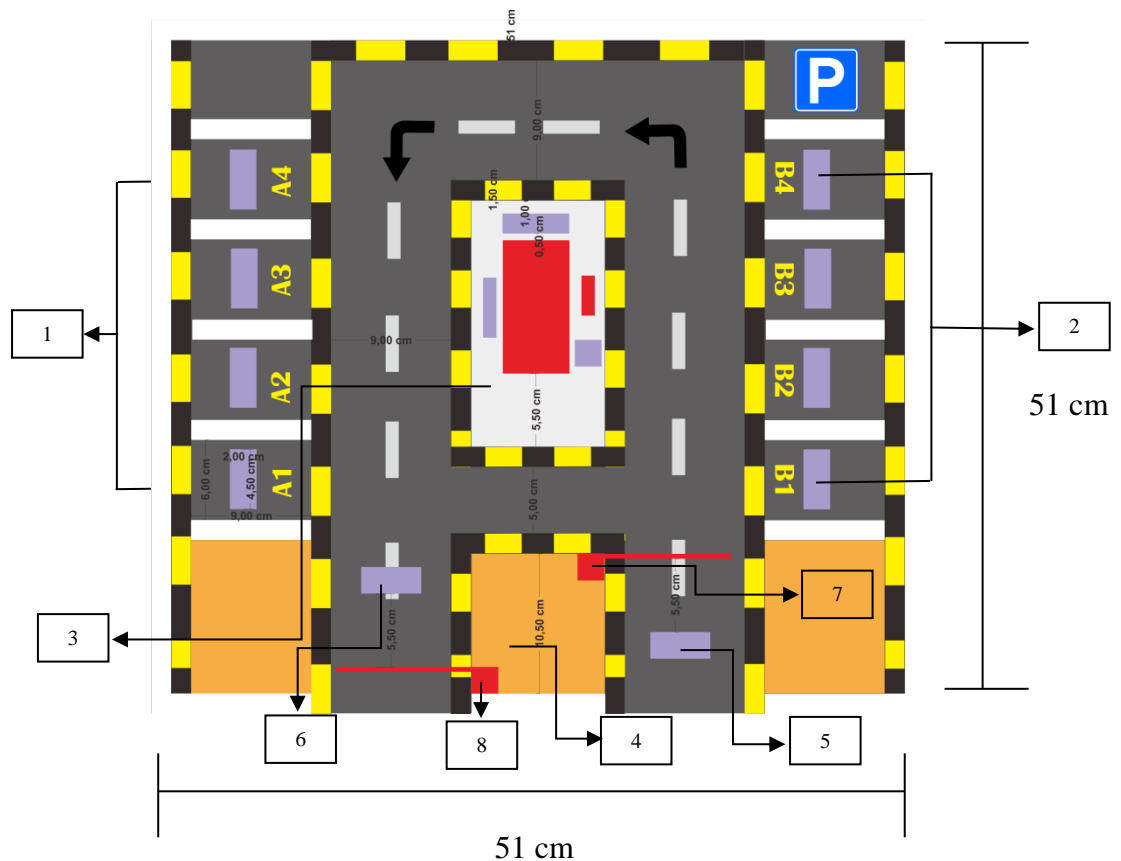
Gambar 4. 2 *Flowchart* cara kerja simulasi sistem *smart parking*

4.2 Perancangan dan Pembuatan *Hardware*

Perancangan dan pembuatan *hardware* terdiri dari dua bagian yaitu bagian mekanik dan elektrik. Berikut tahap-tahap perancangan dan pembuatan *hardware*:

4.2.1 Perancangan *Hardware* Secara Mekanik

Proses perancangan *hardware* secara mekanik pada simulasi sistem *smart parking* yaitu pembuatan konstruksi *base* parkir, berikut adalah tampilan konstruksi *base* parkir sistem *smart parking*.



Gambar 4. 3 Desain *base* sistem *smart parking*

Berikut keterangan dari desain *base* sistem *smart parking*:

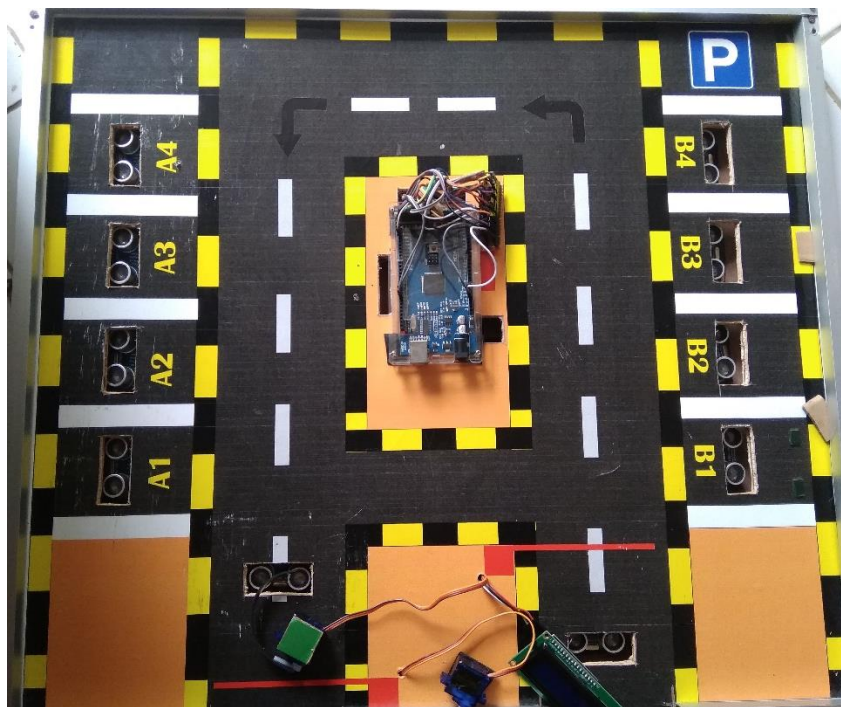
1. Penempatan sensor ultrasonik yang digunakan untuk mendeteksi kendaraan pada slot A1 sampai dengan A4.
2. Penempatan sensor ultrasonik yang digunakan untuk mendeteksi kendaraan pada slot B1 sampai dengan B4.
3. Penempatan sistem kontrol yaitu Arduino Mega 2560.
4. Tempat *monitoring* slot tempat parkir serta pembayaran parkir yang dioperasikan oleh operator.
5. Penempatan sensor ultrasonik untuk mendeteksi kendaraan masuk.

6. Penempatan sensor ultrasonik untuk mendeteksi kendaraan keluar.
7. Penempatan Servo yang digunakan sebagai portal masuk
8. Penempatan Servo yang digunakan sebagai portal keluar

4.2.2 Pembuatan *Hardware* Secara Mekanik

Base pada sistem *smart parking* dibuat dari triplek berbentuk persegi empat dengan panjang 51 cm dan lebar 51 cm. *Base* parkir ini terdiri dari dua bagian yaitu *base* bagian atas untuk menempatkan Arduino Mega 2560 beserta LCD 16x2 I2C dan Servo Tower Pro SG90 kemudian *base* bagian bawah untuk menempatkan sensor ultrasonik dan jalur pengkabelan dimana kedua bagian tersebut saling terhubung.

Pada *base* bagian atas akan diberikan lubang agar sensor ultrasonik yang berada di *base* bagian bawah akan terlihat untuk mendeteksi kendaraan yang ada di atasnya.



Gambar 4. 4 Hasil akhir pembuatan *base* parkir bagian atas

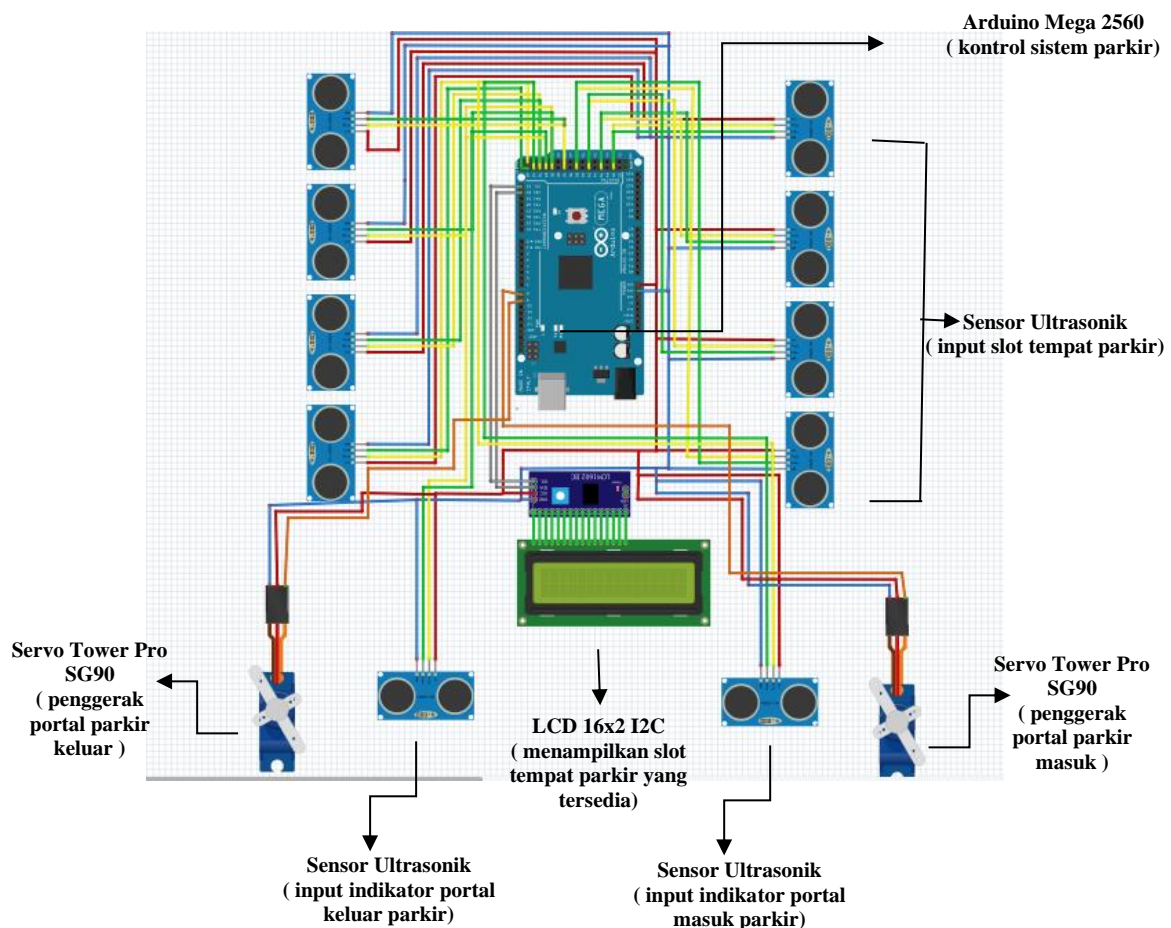
4.2.3 Perancangan dan Pembuatan *Hardware* Secara Elektrik

Pembuatan *hardware* secara elektrik terdiri dari dua bagian yaitu :

1. Membuat skema pengkabelan atau *wiring* dari simulasi sistem *smart parking*.
2. Memasang *wiring* pada komponen.

Berikut penjelasan dari pembuatan *hardware* secara elektrik :

1. Pembuatan skema pengkabelan dibuat menggunakan aplikasi Fritzing. Dimana skema ini bertujuan sebagai rancangan dalam memasang *wiring* agar meminimalisir kesalahan dalam pemasangan *wiring* pada komponen.



Gambar 4. 5 Skematik simulasi sistem *smart parking*

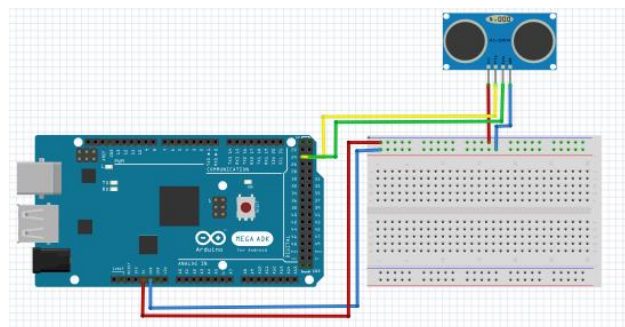
3. Pemasangan *wiring* pada komponen dan *base* bagian bawah dapat dilihat pada Gambar 4. 6.



Gambar 4. 6 Hasil akhir *wiring* pada komponen

4.2.3.1 Perancangan Pembuatan *Hardware* Sensor Ultrasonik

Dalam tahap ini dilakukan perancangan dan pembuatan program untuk sistem kerja dari sensor ultrasonik. Berikut adalah gambar skematik dari percobaan sensor ultrasonik :



Gambar 4. 7 Skematik sensor ultrasonik

Dilihat dari Gambar 4. 7 berikut penjelasan dari pin yang digunakan sensor ultrasonik ke Arduino Mega 2560:

- Pin 5v = Vcc
- Pin 22 = *Trigger*
- Pin 24 = *Echo*
- Gnd = *Gnd*

Setelah dibuatnya skematik maka pembuatan program yaitu sebagai berikut:

```
int trig = 53;           //mendeklarasikan pin 53 sebagai trigger
int echo = 52;          //mendeklarasikan pin 52 sebagai echo
long durasi, jarak;     //variabel durasi dan jarak sensor ultrasonik

void setup() {

pinMode(trig,OUTPUT);   //mengatur triggerpin sensor sebagai output
pinMode(echo,INPUT);    //mengatur echopin sensor sebagai input
Serial.begin (9600);    //mengatur kecepatan pengiriman port serial
}

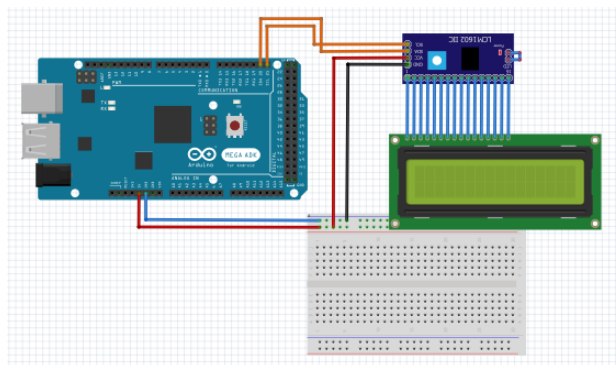
void loop() {

digitalWrite(trig,HIGH); //mengaktifkan trigger sensor ultrasonik
delayMicroseconds(10);  //menunggu selama 10 microseconds
digitalWrite(trig,LOW);  //mematikan trigger sensor ultrasonik
delayMicroseconds(10);  //menunggu selama 10 microseconds
durasi = pulseIn(echo,HIGH); // baca rentan waktu dari trigPin High sampai echoPin high sensor
jarak = (durasi/2) / 29.1; //konversi selang waktu ke CM
Serial.println(jarak);  //mengirimkan variabel data yang ada dalam kurung ke port serial
}
```

Gambar 4. 8 Pemograman sensor ultrasonik

4.2.3.2 Perancangan Pembuatan *Hardware* LCD 16x2 I2C

Pada tahap ini dilakukan perancangan dan pembuatan program LCD untuk mengetahui sistem kerja dari LCD 16x2 I2C. Berikut adalah gambar skematik dari LCD:



Gambar 4. 9 Skematik LCD 16x2 I2C

Berikut pemograman LCD 16x2 I2C menampilkan teks “HELLO WORLD “:

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h> //Library untuk LCD

LiquidCrystal_I2C lcd(0x27,16,2); // set the LCD address to 0x27 for a 16 chars and 2 line display

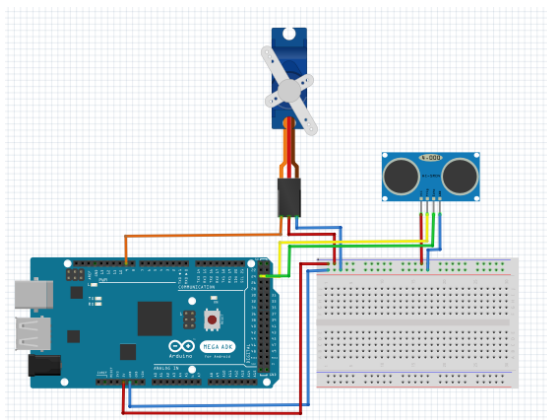
void setup()
{
  lcd.init(); //inisialisasi LCD
  lcd.init(); //inisialisasi LCD
  lcd.backlight(); //menyalakan lampu pada LCD
  lcd.setCursor(1,0); //mengatur kursor ke baris pertama kolom kedua
  lcd.print(" HELLO "); //menampilkan teks pada LCD
  lcd.setCursor(1,1); //mengatur kursor ke baris kedua kolom kedua
  lcd.print(" WORLD "); //menampilkan teks pada LCD
}

void loop()
{
}
```

Gambar 4. 10 Pemograman LCD 16x2 I2C

4.2.3.3 Perancangan Pembuatan *Hardware* Servo Tower SG90

Pada tahap ini dilakukan perancangan dan pembuatan program servo untuk mengetahui sistem kerja dari servo. Namun dalam hal ini menggunakan sensor ultrasonik sebagai pemicu agar servo bergerak, berikut adalah gambar skematik dari servo:



Gambar 4. 11 Skematik Servo Tower Pro SG90

Berikut pemrograman Servo Tower Pro SG90 :

```
#include <Servo.h>           //Library untuk Servo
#define MAX_DISTANCE 500    //jarak maksimal sensor ultrasonik
Servo servo;                //inisialisasi servo

int pinTrigger = 24;        //mengatur trigger sensor pada pin 24
int pinEcho = 25;          //mengatur echo sensor pada pin 25

float durasi, jarak;        //variabel durasi dan jarak pda sensor ultrasonik

void setup()
{
  pinMode(pinTrigger, OUTPUT); //pin trigger sebagai output
  pinMode(pinEcho, INPUT);     //pin echo sebagai input

  Serial.begin(9600);          //kecepatan komunikasi Serial dengan komputer
  servo.attach(9);             //pin PWM 9 untuk servo
}

void loop()
{
  digitalWrite(pinTrigger, HIGH); //mengaktifkan sensor ultrasonik
  delayMicroseconds(10);          //menunggu selama 10 microseconds
  digitalWrite(pinTrigger, LOW);  //mematikan sensor ultrasonik
  durasi = pulseIn(pinEcho, HIGH); //baca rentan waktu dari trigPin High sampai echoPin high sensor
  jarak = (durasi * 0.034) / 2;    //Konversi nilai durasi ke jarak

  if (jarak <= 6) //jarak (cm) dapat diatur dan disesuaikan

  {

    servo.write(45 ); //posisi servo 45 derajat
    delay(3500);      //menunggu selama 3.5 detik
  }
  else
  {

    servo.write(90); //posisi servo 0 derajat
  }
  delay(400);        //menunggu selama 4 microseconds
}
```

Gambar 4. 12 Pemrograman Servo Tower Pro SG90

4.3 Pengujian *Hardware*

Setelah dilakukannya perancangan dan pembuatan *hardware* maka tahap selanjutnya adalah pengujian yang dilakukan untuk melihat apakah sistem kerja dari *hardware* sesuai dengan apa yang diinginkan.

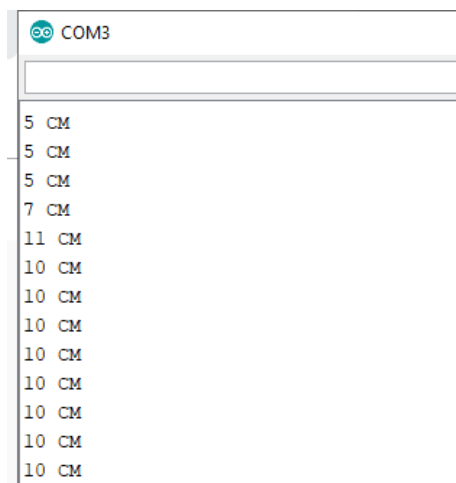
4.3.1 Pengujian Sensor Ultrasonik

Pengujian ini dilakukan menggunakan Arduino Mega 2560 untuk menghubungkannya dan menempatkan sebuah benda di hadapan sensor ultrasonik dengan jarak tertentu untuk melihat hasil pengukuran sensor dan mencocokkannya

dengan hasil sebenarnya. Berikut tabel hasil pengujian pembacaan jarak sensor dengan jarak sebenarnya:

Tabel 4. 2 Pengujian jarak sensor ultrasonik

| Jarak sebenarnya | Percobaan ke | Jarak baca senso |
|------------------|--------------|------------------|
| 5 cm | 1 | 5 cm |
| 5 cm | 2 | 5 cm |
| 5 cm | 3 | 5 cm |
| 10 cm | 4 | 11 cm |
| 10 cm | 5 | 10 cm |
| 10 cm | 6 | 10 cm |



Gambar 4. 13 Hasil pembacaan jarak pada sensor

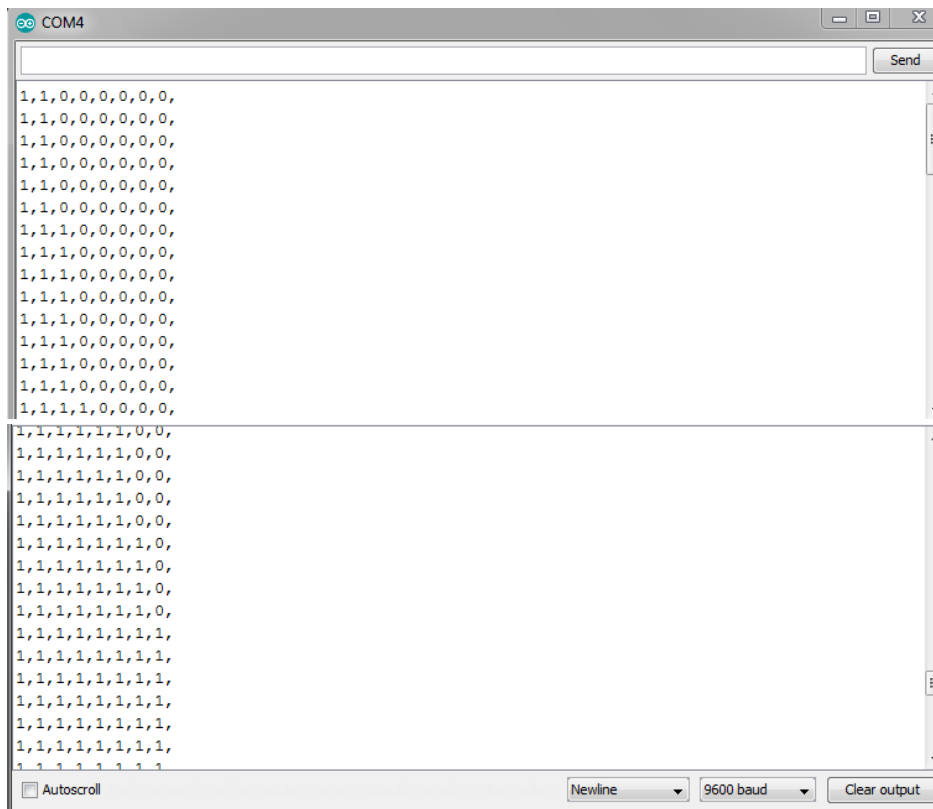
Pada hasil didapat dari pengujian jarak hasil baca sensor dengan jarak sebenarnya terlihat akurat. Namun pada saat menempatkan benda harus benar-benar pas agar jarak yang terukur sama.

Setelah dilakukan pembacaan jarak maka selanjutnya adalah pembacaan keberadaan benda dalam hal ini adalah mobil terhadap sensor. Dimana di dalam program telah ditentukan batas jarak antara sensor dengan mobil adalah minimal 10 cm. jadi jika mobil berada minimal 10 cm dari sensor maka sensor bernilai satu dan jika mobil berjarak lebih dari 10 cm dari sensor maka sensor bernilai 0 atau

dianggap tidak ada kendaraan yang menempati slot parkir. Berikut hasil pengujiannya:

Tabel 4. 3 Pengujian data sensor

| Slot A1 | Slot A2 | Slot A3 | Slot A4 | Slot B1 | Slot B2 | Slot B3 | Slot B4 | Data sensor (A1 s/d B4) |
|---------|---------|---------|---------|---------|---------|---------|---------|-------------------------|
| ✓ | ✓ | | | | | | | 1,1,0,0,0,0,0,0, |
| ✓ | ✓ | ✓ | | | | | | 1,1,1,0,0,0,0,0, |
| ✓ | ✓ | ✓ | ✓ | | | | | 1,1,1,1,0,0,0,0, |
| ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | | 1,1,1,1,1,1,0,0, |
| ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | 1,1,1,1,1,1,1,0, |
| ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | 1,1,1,1,1,1,1,1, |



Gambar 4. 14 Pengujian baca data sensor ultrasonik

4.3.2 Pengujian LCD 16x2 I2C

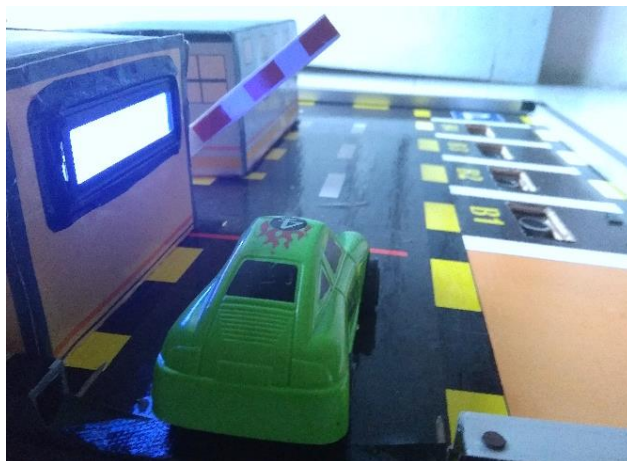
Pada proyek akhir ini LCD 16x2 I2C akan digunakan untuk menginformasikan kepada pengendara mobil jumlah *area* parkir yang tersedia dan juga menginformasikan jika *area* parkir tersebut penuh. Berikut hasil pengujian LCD menampilkan sebuah teks.



Gambar 4. 15 Hasil pengujian LCD 16x2 I2C

4.3.3 Pengujian Servo Tower Pro SG90 dengan Sensor Ultrasonik

Servo Tower Pro SG90 digunakan sebagai portal di *area* parkir. Pada proyek akhir ini menggunakan dua buah Servo Tower Pro SG90 yang berada di *area* masuk parkir dan *area* keluar parkir. Dalam pengujian servo juga menggunakan sensor ultrasonik sebagai pemicu untuk servo bergerak. Berikut hasil pengujian servo dengan sensor ultrasonik dimana servo akan bergerak jika sensor ultrasonik mendeteksi benda :



Gambar 4. 16 Hasil pengujian Servo Tower Pro SG90

4.3.4 Pengujian sensor Ultrasonik dengan LCD 16x2 I2C

Pengujian dilakukan menggunakan 8 *area* parkir (A1, A2, A3, A4, B1, B2, B3, B4). Pada tampilan LCD menginformasikan jumlah *area* parkir yang tersedia, jika sensor mendeteksi mobil di *area* parkir tertentu maka *area* parkir yang tersedia akan berkurang. Jika *area* parkir penuh maka tampilan pada LCD akan menginformasikan bahwa slot parkir penuh.

Tabel 4. 4 Pengujian sensor ultrasonik dengan LCD

| Tampilan LCD (<i>Free Space</i>) | Slot A1 | Slot A2 | Slot A3 | Slot A4 | Slot B1 | Slot B2 | Slot B3 | Slot B4 |
|---------------------------------------|------------|------------|------------|------------|------------|------------|------------|------------|
| 8 | | | | | | | | |
| 7 | | | | | | ✓ | | |
| 6 | | | | | | ✓ | | ✓ |
| 5 | ✓ | ✓ | | | ✓ | | | |
| 4 | ✓ | | ✓ | ✓ | | ✓ | | |
| 3 | ✓ | ✓ | ✓ | ✓ | ✓ | | | |
| 2 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | |
| 1 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | |
| 0 (parkir penuh) | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |



Gambar 4. 17 Pengujian sensor ultrasonik dengan LCD



Gambar 4. 18 Keadaan LCD jika parkir penuh

4.4 Perancangan dan Pembuatan *Software*

Pada perancangan dan pembuatan *software* maka akan dibahas perancangan dan pembuatan *interface monitoring* slot tempat parkir serta sistem pembayaran parkir menggunakan aplikasi Microsoft Visual Studio dan aplikasi Microsoft Access.

4.4.1 Perancangan dan Pembuatan *Database*

Microsoft Access adalah aplikasi yang digunakan untuk membuat *database* atau basis data. *Database* adalah kumpulan data berbentuk tabel. Data yang digunakan akan diolah menjadi informasi. Berikut adalah *database* yang dibuat :

| No_karcis | Jam_Masuk | Tgl_Masuk | Slot_A1 | Slot_A2 | Slot_A3 | Slot_A4 | Slot_B1 | Slot_B2 | Slot_B3 | Slot_B4 |
|-----------|-----------|------------|---------|---------|---------|---------|---------|---------|---------|---------|
| B1001 | 20:16:04 | 2020/08/17 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| b1002 | 20:34:55 | 2020/08/17 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |

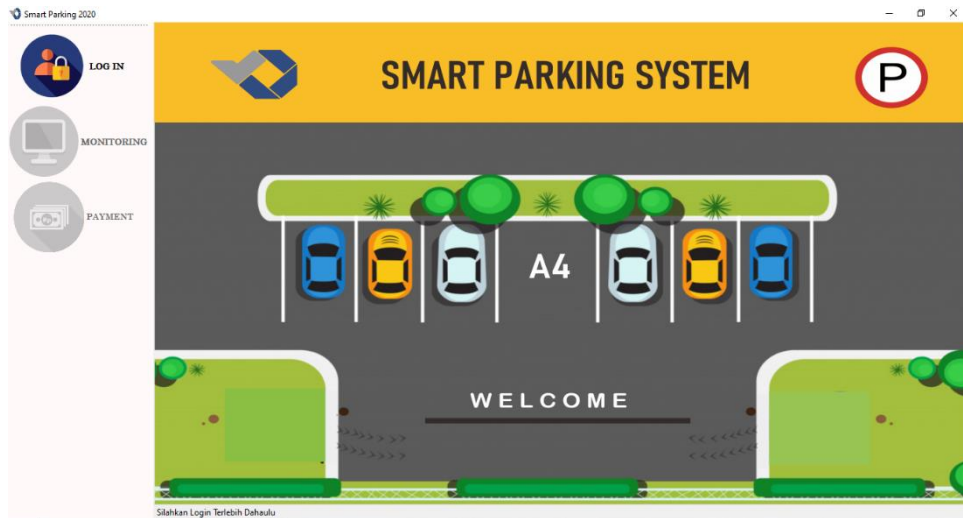
Gambar 4. 19 Database Ms.Access

Untuk mengolah *database* agar tekoneksi dengan Visual Studio maka *database* ini harus disimpan di dalam folder *debug* dari *project* Visual Studio yang telah dibuat.

4.4.2 Perancangan dan Pembuatan *Interface* Microsoft Visual Studio

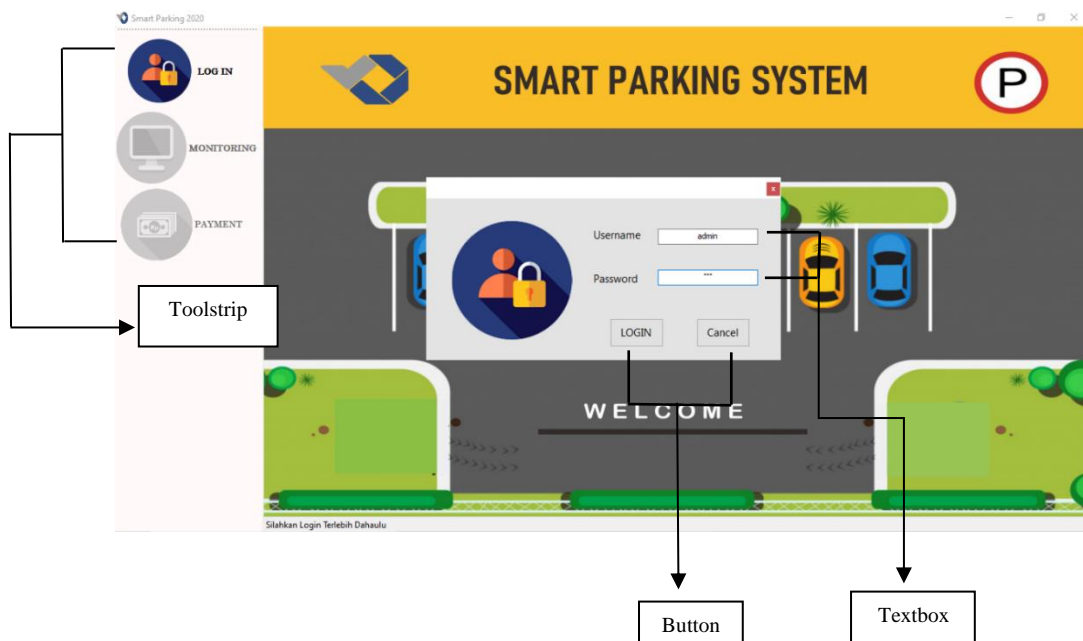
Interface menggunakan aplikasi Microsoft Visual Studio dirancang dalam 3 *tab* yang terdiri dari pertama *tab* menu utama yaitu mencakup menu-menu yang dapat dipilih sesuai keinginan operator yaitu ada menu *login* dan *logout* dari sistem kemudian menu *monitoring* dan menu *payment*, kedua yaitu *tab monitoring* yang mencakup informasi mengenai slot parkir kosong dan terisi yang ditandai dengan warna. Ketiga yaitu *tab payment* yaitu tampilan sistem bayar parkir dimana operator dapat melakukan *input* data dari nomor karcis yang ditempati kemudian akan ditampilkan biaya parkir. Kemudian di dalam menu *payment* juga terdapat tampilan *database* dari slot tempat parkir. Berikut tampilan dari *interface* yang dibuat pada Microsoft Visual Studio 2012 :

1. Menu Utama



Gambar 4. 20 Tab menu utama

Pada bagian menu utama terdapat menu *login*, *monitoring* dan *payment*. Namun pada bagian menu *monitoring* dan *payment* belum bisa diakses karena untuk mengakses sistem diharuskan untuk melakukan *login* terlebih dahulu.



Gambar 4. 21 Menu utama dan *login*

Berikut keterangan dari *interface* menu utama dan *login*:

- *Toolstrip*, digunakan untuk menempatkan beberapa menu di dalamnya yaitu *login*, *monitoring* dan *payment*. Maka untuk masuk ke sistem klik menu *login* maka akan muncul jendela *login* yang diperintahkan untuk mengisi *username* dan *password*.
- *Textbox*, digunakan untuk menuliskan *username* dan *password* yang telah ditentukan di dalam program.
- *Button*, terdiri dari dua *button* yaitu *button login* untuk masuk ke sistem dan *button cancel* untuk menutup jendela *login*.

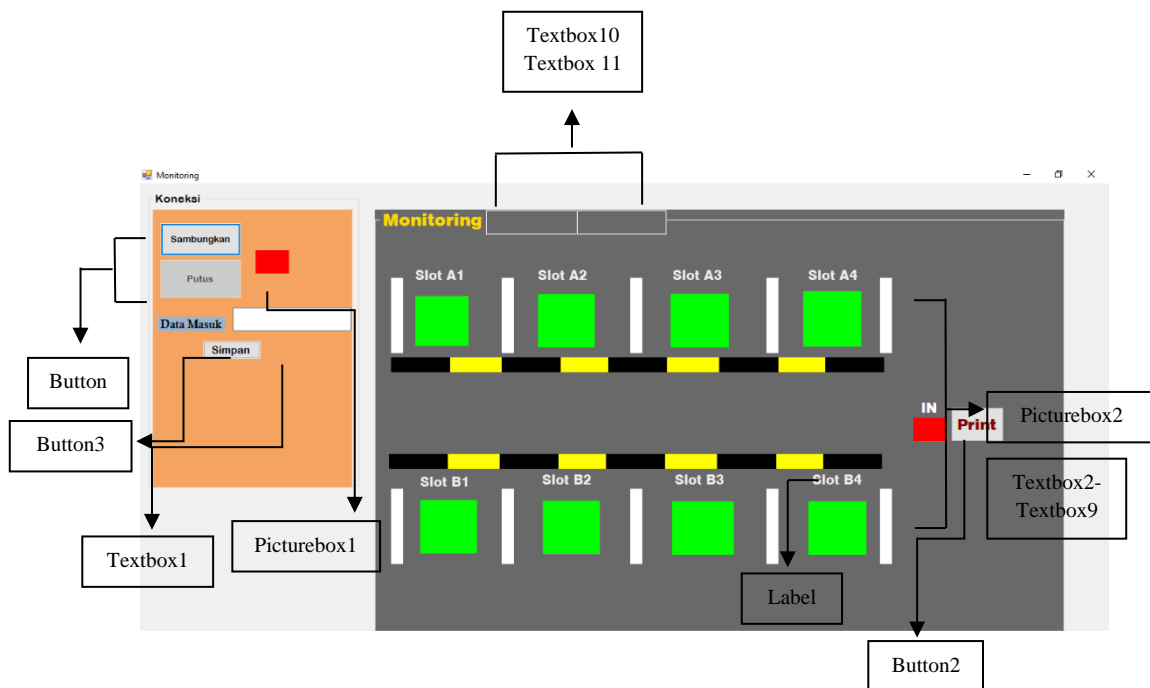


Gambar 4. 22 Akses menu utama

Setelah melakukan *login* maka masuk ke sistem dan dapat mengakses *monitoring* slot tempat parkir dan pembayaran parkir yaitu pada Gambar 4. 22.

2. Menu *monitoring*

Pada menu *monitoring* terdapat informasi slot tempat parkir yang kosong dan terisi yang memungkinkan operator dapat memantau slot tempat parkir dan memantau kendaraan yang ingin masuk. Berikut tampilan *interface* dari menu *monitoring*:



Gambar 4. 23 Menu *monitoring*

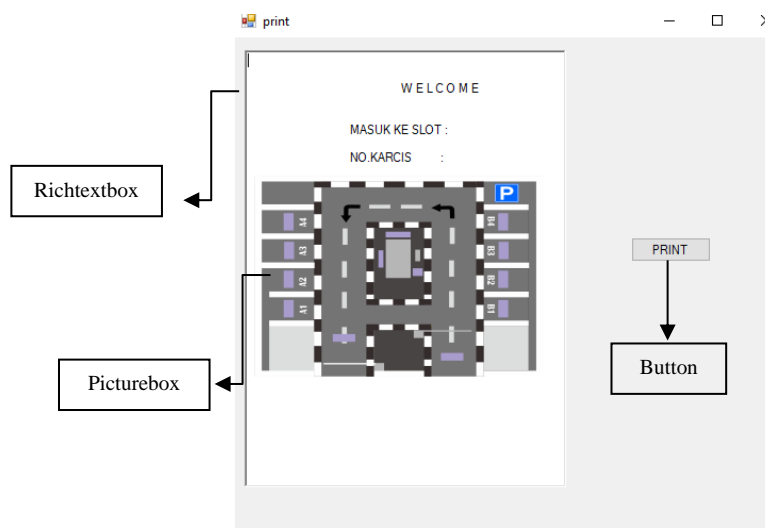
Dalam menu *monitoring* terdapat *button* sambungkan dimana untuk memulai *monitoring* maka harus menekan *button* sambungkan agar terhubung dengan Arduino. Berikut keterangan dari *interface* menu *monitoring*:

- *Button*, terdiri dari *button* sambungkan dan putus. *Button* sambungkan digunakan untuk menghubungkan Arduino dengan Visual Studio dan *button* putus digunakan untuk menghentikan komunikasi hubungan antara Arduino dengan sistem.
- *Picturebox1*, berfungsi sebagai indikator yang akan berubah warna menjadi warna hijau jika komunikasi Arduino dengan sistem telah terhubung.
- *Textbox1*, digunakan untuk menampilkan data slot tempat parkir masuk dan keluar secara keseluruhan.
- *Textbox2-Textbox9*, ditempatkan di layer bawah *picturebox2* yang digunakan untuk menampilkan data slot tempat parkir masuk dan keluar secara terpisah, dalam hal ini data yang masuk pada *textbox1* akan dipisah disini.
- *Textbox10, Textbox11* digunakan untuk tampilan waktu dan tanggal.

- *PictureBox2*, pada label slot A1 sampai dengan slot B4 terdapat *picturebox2* di masing-masing slot dimana *picturebox2* ini terhubung dengan *textbox2-textbox9* yang ada di bawahnya, jika *textbox2-textbox9* menunjukkan data 1 maka *picturebox2* akan menunjukkan warna merah yang artinya slot tersebut telah terisi dan jika menunjukkan data 0 maka *picturebox2* akan tetap berwarna hijau yang menandakan slot masih tersedia.
- *Button 3*, digunakan untuk menyimpan data slot parkir beserta waktu dan tanggal ke *database*.
- *Button 2*, digunakan untuk menampilkan *form print* karcis.

3. Menu *Print*

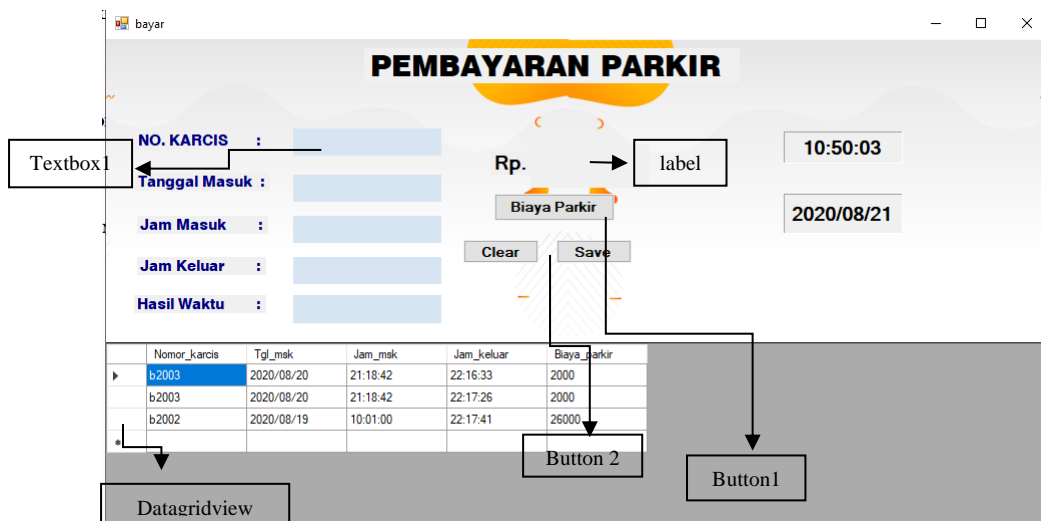
Pada menu *print* terdapat komponen *richtextbox* untuk memasukkan slot parkir yang harus diisi serta *button print* yang digunakan untuk mencetak tulisan yang ada di dalam *richtextbox* serta gambar yang ada pada *picturebox*.



Gambar 4. 24 Menu *print*

4. Menu *payment*

Pada menu *payment* yaitu mencakup sistem bayar parkir dan tampilan *database* slot parkir masuk.



Gambar 4. 25 Menu pembayaran parkir

Berikut keterangan dari *interface* menu pembayaran parkir:

- *Datagridview*, digunakan untuk menampilkan *database* dari Ms. Access.
- *Textbox1*, pada No.Karcis digunakan untuk memanggil *database* yang ada pada Ms. Access yang kemudian akan menampilkan data pada tanggal masuk, jam masuk, jam keluar dan jumlah jam parkir.
- *Button 1*, digunakan untuk memanggil biaya parkir.
- *Button 2*, digunakan untuk menghapus teks yang ada pada *textbox*.
- *Label*, digunakan untuk menampilkan biaya parkir.

4.4.2.1 Komunikasi Serial pada Microsoft Visual Studio 2012

Tahap awal pembuatan *interface* adalah membuat komunikasi serial antara Arduino dengan Visual Studio, untuk membuatnya maka dilakukan pemrograman terlebih dahulu. Data dari Arduino dapat dibaca di dalam Visual Studio, seperti yang telah dijelaskan pada menu *monitoring*. Komponen yang digunakan adalah *button* dengan teks sambungkan.

Tambahkan komponen *Serialport* di dalam Visual Studio dan atur pada *properties* com dan *baudrate* sesuai dengan Arduino yang terpasang. Kemudian *coding* pada *button* sambungkan sebagai berikut:

```

Sub arduinoaktif() ' procedure hubungan serial (Arduino) aktif
SerialPort1.Open() ' hubungan serial aktif
Try
    If SerialPort1.IsOpen() Then 'jika hubungan serial aktif
        PB_connect.BackColor = Color.Green ' maka warna dari picturebox akan menjadi warna hijau
        Call aktif() ' panggil sub aktif
        Timer1.Start() ' mengaktifkan timer
        Timer2.Stop() ' timer 2 belum aktif
    End If
Catch ex As Exception ' menangkap kesalahan yang mungkin terjadi dimana ex adalah variabel
    MessageBox.Show(" Tidak Terhubung ", "Note") ' maka akan menampilkan message box
End Try
End Sub

Private Sub btn_connect_Click(sender As Object, e As EventArgs) Handles btn_connect.Click 'jika tombol sambungkan di klik
    Call arduinoaktif() ' memanggil sub arduinoaktif
End Sub

```

Gambar 4. 26 Pemrograman komunikasi Arduino dengan Visual Studio

Pada Gambar 4. 26 terdapat perintah `If SerialPort1.IsOpen() Then PB_connect.BackColor = Color.Green` artinya jika koneksi terhubung maka indikator warna akan berubah menjadi warna hijau.

4.4.2.1.1 Uji Coba Koneksi Arduino dengan Visual Studio

Pengujian ini dilakukan setelah pembuatan program. Berikut hasil pengujian komunikasi Arduino dengan Visual Studio:

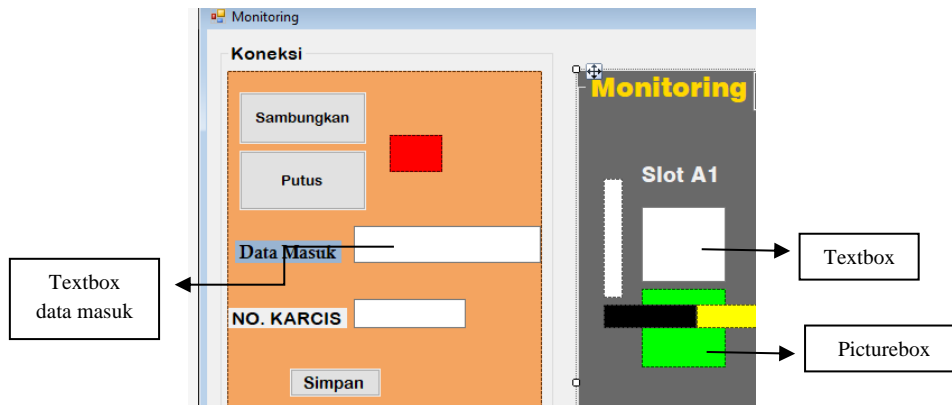


Gambar 4. 27 Koneksi Arduino dengan Visual Studio

Koneksi Arduino dengan Visual Studio ditandai dengan indikator warna yang terdapat di samping tombol sambungkan. Jika berwarna hijau maka koneksi telah terhubung dan pada data masuk pun akan tampil data keseluruhan slot.

4.4.2.2 Pemisahan Data Sensor di Visual Studio

Tahap kedua pembuatan *interface* adalah melakukan pemisahan data sensor. Data sensor dari Arduino secara keseluruhan yang ada pada data masuk akan dipisah ke dalam *textbox* pada masing-masing slot tempat parkir seperti pada Gambar 4. 27.



Gambar 4. 28 *Textbox* pemisahan data

Maka setelah itu diberikan pemrograman sebagai berikut :

1. Tahapan diberikan inialisasi variabel terlebih dahulu

```
Public Class formmonitoring
    Private bacadata As String = String.Empty
    Private Bytenumber As Integer 'mendeklarasikan variabel di dalam satu module dengan tipe data bilangan bulat
    Private BytetoRead As Integer
    Private byteEnd(2) As Char 'variabel yang menampung 1 buah karakter
    Dim inputdata As String 'variabel sebagai karakter atau teks
    Dim prosesoff As Boolean = False 'variabel utk nilai logika
    Dim disconnect As Boolean = False
    Dim data(8) As String 'jumlah data yang masuk dari arduino
```

Gambar 4. 29 Inialisasi variabel

2. Proses pemisahan data

Dalam proses pemisahan data, data dari serial dari Arduino akan diterima dan masuk ke dalam *textbox* data masuk yang kemudian akan dipisah ke dalam masing-masing *textbox* pada slot yang dimulai dari *textbox* untuk slot A1 sampai slot B4.

```

Private Sub proses_fix()
inputdata = TB_datamasuk.Text ' string inputdata = textbox
Dim panjang_data As Integer ' variabel panjang_data tipe data integer( bil.bulat)
Dim x As Integer
Dim z As Integer
inputdata = TB_A1.Text
panjang_data = Len(TB_A1.Text) 'menghitung jumlah karakter (panjang data) dari string tb_A1

Dim i As Integer ' variabel i tipe data integer
i = 0
z = 0

For x = 1 To Len(bacadata$) 'looping untuk data ke dua sampai terakhir
If Mid(bacadata$, x, 1) = "," Then
z = z + 1 'jika mendeteksi "," ke dua dimulai ke x dengan panjang data 1, jadi x sebagai simbol dari "|"
data(i) = Mid(bacadata$, z, x - z) 'data ke i = integer dari pengambilan data TB_datamasuk dimulai dari z(awalnya 2) sampai ke x - z
i = i + 1 'i increment dari 0 ke satu, satu ke dua dst...
z = x 'z di increment dengan x karena panjang max = x dan awal data = z maka untuk mendapatkan data = x-z
End If
Next x
End Sub

```

Gambar 4. 30 Pemograman pemisah data

3. Proses *input* data ke dalam *textbox* masing-masing label slot parkir. Dimana TB_datamasuk membaca data serial dari Arduino secara keseluruhan dan untuk TB_A1 sampai dengan TB_B4 akan menerima data per sensor di masing-masing slot dan jika slot bernilai 1 maka *picturebox* pada masing-masing slot akan berubah menjadi warna merah.

```

Sub datasensor()
    If SerialPort1.IsOpen Then ' jika hubungan serial aktif
        TB_datamasuk.Text = bacadata 'pada textbox data masuk akan membaca data serial yang masuk secara keseluruhan
        TB_A1.Text = data(0) ' textbox A1 sampai dengan B4 akan menerima pecahan data dari serial yang masuk di data masuk dengan
        TB_A2.Text = data(1) ' menggunakan variabel data sebagai string untuk memisahkan data
        TB_A3.Text = data(2)
        TB_A4.Text = data(3)
        TB_B1.Text = data(4)
        TB_B2.Text = data(5)
        TB_B3.Text = data(6)
        TB_B4.Text = data(7)
        TB_IN.Text = data(8)
    End If
End Sub

Private Sub Timer1_Tick(sender As Object, e As EventArgs) Handles Timer1.Tick 'jika timer aktif
    TB_jam.Text = Format(Now, "HH:mm:ss") ' textbox untuk jam akan diisi dengan format waktu yang ada pada komputer
    TB_tgl.Text = Format(Now, "yyyy/MM/dd") ' ' textbox untuk tanggal akan diisi dengan format waktu yang ada pada komputer

    Call datasensor() ' memanggil sub datasensor yang telah dibuat sebelumnya
    Call warna_A1() ' memanggil sub warna yang telah dibuat
    Call warna_A2()
    Call warna_A3()
    Call warna_A4()
    Call warna_B1()
    Call warna_B2()
    Call warna_B3()
    Call warna_B4()
    Call warna_IN()
End Sub

Sub warna_A1() ' procedure untuk perubahan warna pada picture box
    If data(0) = 1 Then ' jika data seria yang masuk adalah 1 maka
        PB_A1.BackColor = Color.Red ' warna picture box akan menjadi warna merah
    Else ' jika tidak
        PB_A1.BackColor = Color.Lime ' warna picture box akan menjadi warna hijau
    End If
End Sub

Sub warna_A2()
    If data(1) = 1 Then
        PB_A2.BackColor = Color.Red
    Else
        PB_A2.BackColor = Color.Lime
    End If
End Sub

Sub warna_A3()
    If data(2) = 1 Then
        PB_A3.BackColor = Color.Red
    Else
        PB_A3.BackColor = Color.Lime
    End If
End Sub

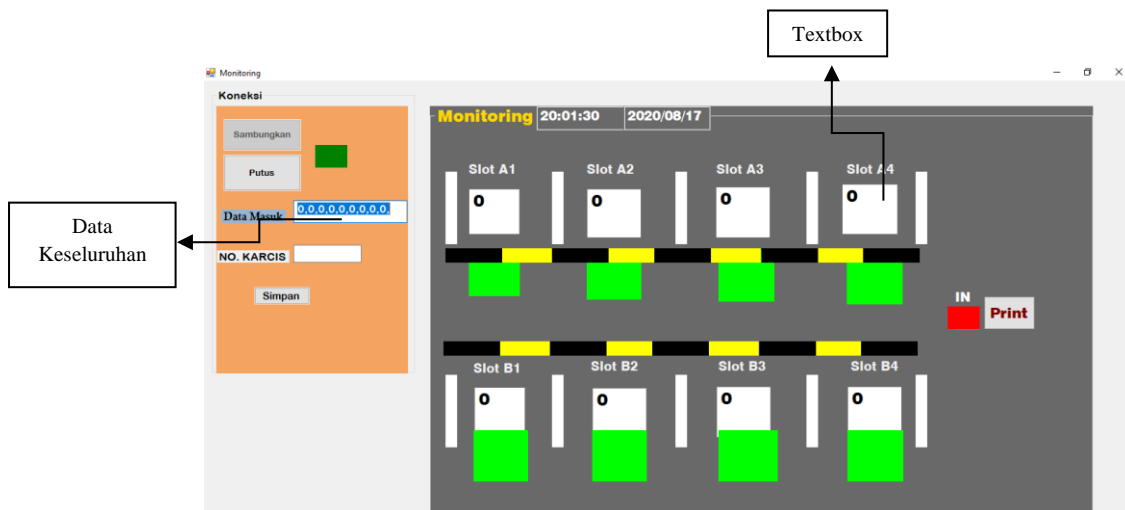
Sub warna_A4()
    If data(3) = 1 Then
        PB_A4.BackColor = Color.Red
    Else
        PB_A4.BackColor = Color.Lime
    End If
End Sub

```

Gambar 4. 31 Pemograman *input* data Arduino

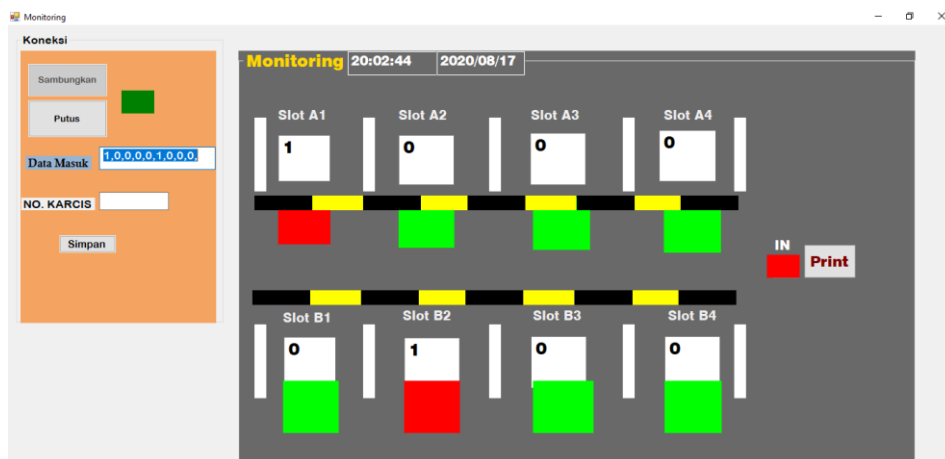
4.4.2.2.1 Uji Coba Pemisahan Data

Dalam uji coba pemisahan data ini, data sensor secara keseluruhan ada data masuk akan dipisah di dalam *textbox* masing-masing slot. Dalam hal ini nilai yang tampil adalah 0 dan 1, dimana 0 menandakan slot tersedia dan 1 menandakan slot telah terisi.



Gambar 4. 32 Pemisahan data ke dalam *textbox*

Kemudian jika terjadi perubahan data contohnya pada Gambar 4. 32 data slot B2 dan A1 bernilai 1 yang menandakan slot telah terisi dan otomatis indikator warna akan berubah yaitu berwarna merah yang juga menandakan bahwa slot telah terisi.



Gambar 4. 33 Perubahan indikator warna pada *picturebox*

4.2.4.3 Koneksi Visual Studio dengan *Database*

Tahap ketiga adalah koneksi Visual Studio dengan *database*. Pada pemrograman ini *database* yang telah dibuat didalam Ms.Access akan ditampilkan di dalam Visual Studio. Kemudian juga dilakukan *input* data dari sensor yang ada di masing-masing slot parkir ke dalam *database*.

1. Membuat *database* pada Ms. Access terlebih dahulu
2. Buat *module* pada pemrograman Visual Studio untuk koneksi antara *database* dengan Visual Studio. Berikut pemrograman *module*:

```
Imports System.Data.OleDb 'Mendaftarkan component untuk akses data menggunakan OLEDB data provider

Module Module1
    Public conn As OleDbConnection ' mendeklarasikan variabel yang akan digunakan koneksi ke database
    Public da As OleDbDataAdapter ' mendeklarasikan variabel yang akan digunakan menghubungkan database dengan dataset
    Public cmd As OleDbCommand ' mendeklarasikan variabel yang akan digunakan melaksanakan perintah sql query
    Public dr As OleDbDataReader ' mendeklarasikan variabel yang akan digunakan membaca data read only
    Public ds As DataSet ' mendeklarasikan variabel yang akan digunakan tempat tampungan data output dataadapter

    Sub konek() 'procedure engan variabel konek
        Try ' error handling
            conn = New OleDbConnection("provider=microsoft.ace.OleDb.12.0; data source=arduino_db.accdb") ' membuat koneksi ke database dengan provider oledb dan sumber data
            conn.Open() ' variabel conn ( koneksi ke database ) akan terbuka
        Catch ex As Exception
            MsgBox(ex.Message) ' menampilkan error
        End Try
    End Sub
End Module
```

'Modul merupakan bagian yang sengaja dipisahkan untuk memudahkan pemrograman. Dalam module dapat dimasukkan procedure dan fungsi dan kemudian digunakan oleh beberapa form.

Gambar 4. 34 Pembuatan *module* koneksi

Karena *database* yang digunakan menggunakan Ms.Access maka digunakan aplikasi pemrograman *interface* pengolahan *database* yaitu OLEDB. Dengan menggunakan *sub procedure* konek dengan variabel conn untuk memasukkan *provider* dari *database* serta *database* yang telah dibuat yaitu arduino_db.accdb.

3. Menampilkan *database* pada *form* di Visual Studio. Dalam hal ini menggunakan komponen *datagridview* untuk menampilkan *database*. Berikut pemrogramannya:

```
Sub tampil() ' procedure untuk menampilkan database pada datagridview
    Call konek() ' memanggil koneksi ke database
    da = New OleDbDataAdapter("select * from data_parkir_keluar", conn) 'membaca data yang kemudian menghubungkan dengan dataset
    ds = New DataSet 'membuat dataset baru
    ds.Clear()
    da.Fill(ds, "data_parkir_keluar") ' mengisi dataset dengan data yang dibaca di datadapter
    DataGridView1.DataSource = ds.Tables("data_parkir_keluar") ' mengisi datagridview dengan dataset tabel dengan sumber data dari data_parkir_keluar
    DataGridView1.ReadOnly = True ' datagridview hanya bertugas membaca
End Sub
```

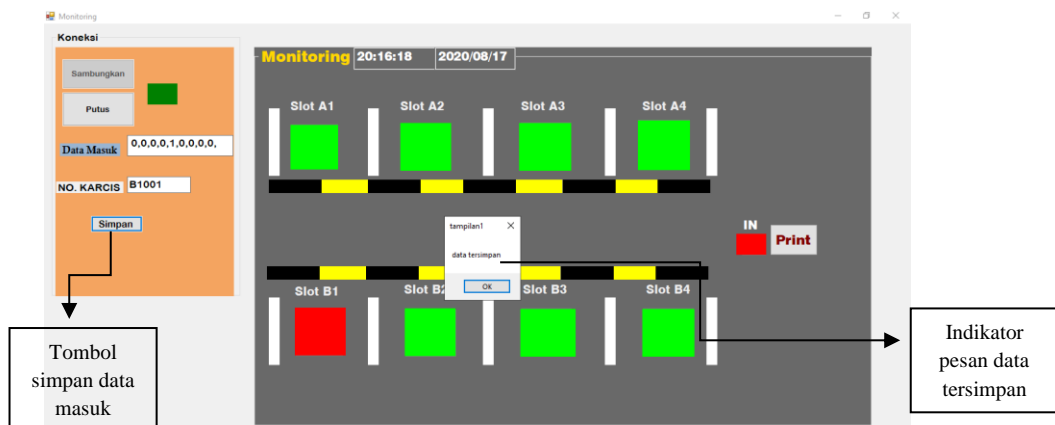
Gambar 4. 35 Menampilkan *database* ke dalam *datagridview*

Dalam pemrograman diatas untuk menampilkan *database* pada *datagridview* yaitu dengan memanggil *sub* konek pada *module* yang telah dibuat,

kemudian pada OleDbDataAdapter masukkan perintah pilih semua data dari tabel yang bernama data dalam *database* arduino_db. Kemudian set data tersebut dengan menginputkannya dalam *datagridview*. Panggil *sub* tampil pada saat *form* dijalankan maka *database* pun akan tampil.

4.2.4.3.1 Uji Coba Koneksi dengan Database

Pada uji coba koneksi Visual Studio dengan *database* operator akan menginputkan nomor karcis sesuai format karcis yaitu slot yang terisi dan nomor masuk. Pada Gambar 4. 36 slot yang terisi adalah slot B1 maka operator akan menginputkan nomor karcis B1001. Kemudian simpan data ke dalam *database* dengan menekan tombol simpan. Maka akan muncul pesan yang menandakan data tersimpan.



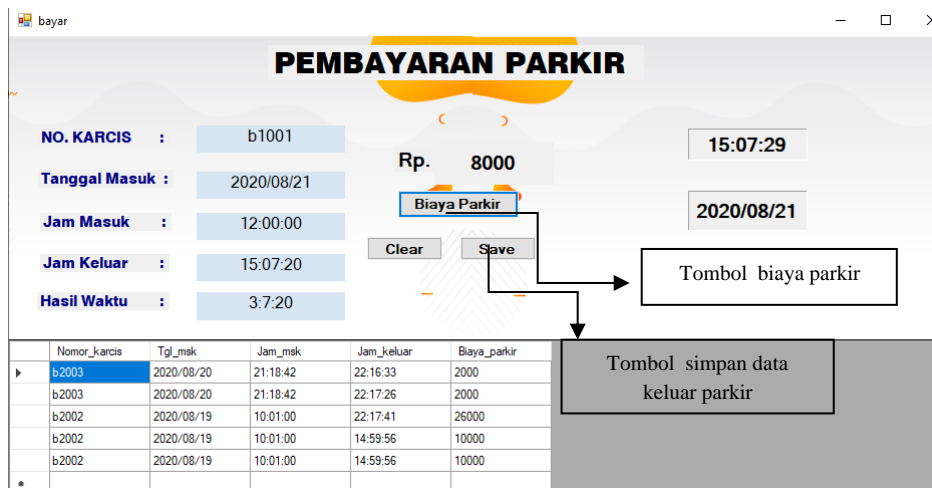
Gambar 4. 36 Simpan data ke dalam *database*

Kemudian pada menu pembayaran akan dilakukan pemanggilan data dari nomor karcis yang telah di tersimpan di dalam *database*.



Gambar 4. 37 Tampilan *database* di Visual Studio

Kemudian setelah dilakukan pemanggilan nomor karcis maka otomatis akan muncul data dari nomor karcis tersebut berupa tanggal masuk, jam masuk, jam keluar serta jumlah jam parkir kendaraan. Untuk mengetahui biaya parkir maka tekan tombol biaya parkir maka otomatis biaya parkir akan tampil. Kemudian simpan data keluar ke dalam *database* dengan menekan tombol *save*



Gambar 4. 38 Tampilan biaya parkir

4.4.2.4 Koneksi Visual Studio dengan *Printer*

Tahap terakhir adalah koneksi Visual Studio dengan *printer*. Pada pemrograman akan dibuat bagaimana Visual Studio terhubung dengan *printer* yang digunakan untuk mencetak karcis parkir. Berikut pemrogramannya:

```
Public Class print
    Private Sub PrintDocument1_PrintPage(sender As Object, e As Printing.PrintPageEventArgs) Handles PrintDocument1.PrintPage
        Dim font1 As New Font("arial", 20, FontStyle.Regular) ' mengatur font style, size huruf dan jenis
        e.Graphics.DrawString(RichTextBox1.Text, font1, Brushes.Black, 100, 100) ' setting teks yang akan di print pada richtextbox, jenis tulisan font1, warna teks, dan Koordinat
        e.Graphics.DrawImage(PictureBox1.Image, 5, 5) ' mensetting gambar dengan Koordinat x di pojok kiri atas teks yang digambar, Koordinat y di pojok kiri atas teks yang digam
    End Sub

    Private Sub btn_print_Click(sender As Object, e As EventArgs) Handles btn_print.Click ' jika tombol print diklik
        PrintDocument1.Print() ' perintah print document
    End Sub
End Class
```

Gambar 4. 39 Pemrograman koneksi *printer* dengan Visual Studio

Pada program diatas menggunakan komponen *printdocument* yang ada pada Visual Studio yang difungsikan untuk perintah *print* serta menggunakan *richtextbox* untuk menuliskan slot tempat parkir yang dituju. Fungsi *print* ini akan dikerjakan jika tombol ditekan.

4.4.2.4.1 Uji Coba *Print* Karcis

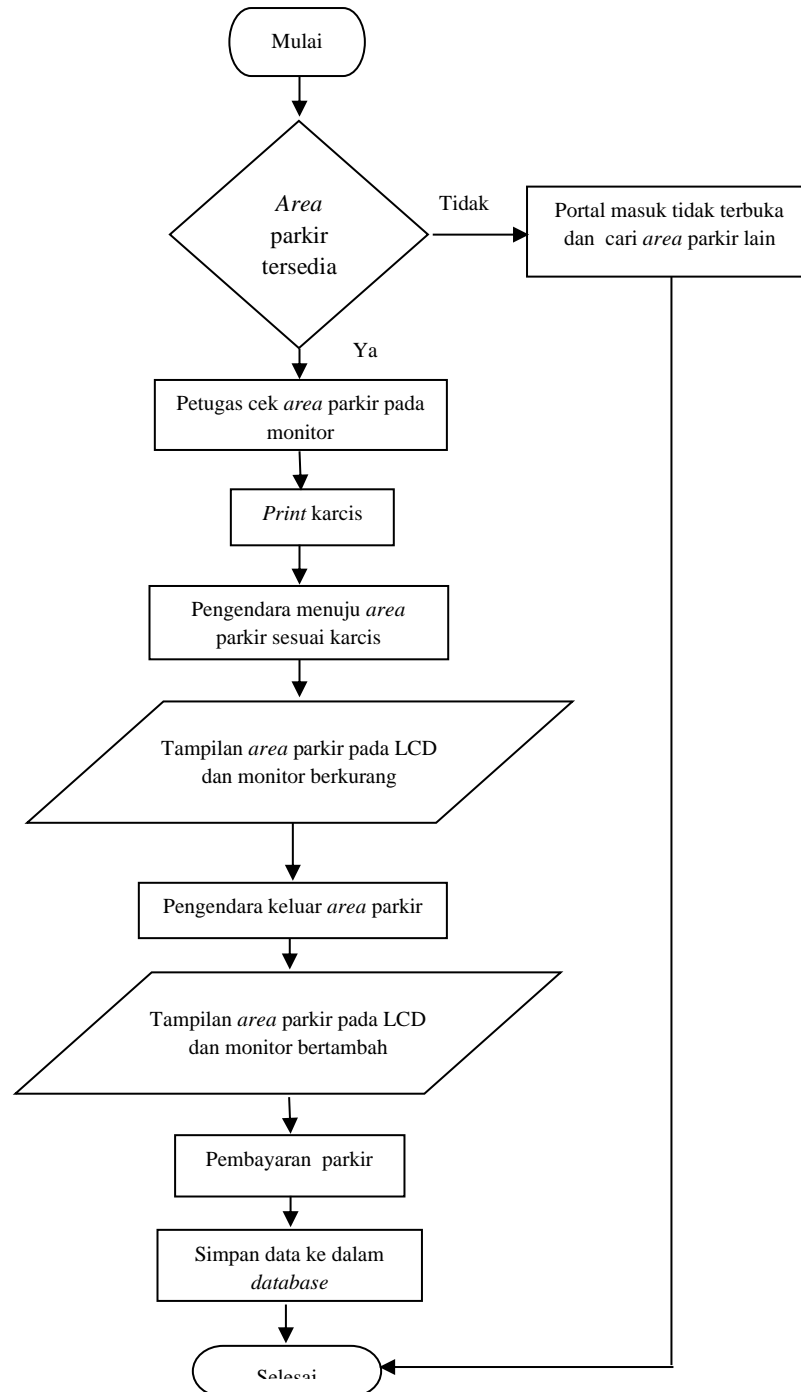
Berikut hasil dari pengujian *print* melalui Visual Studio:



Gambar 4. 40 Hasil *print* karcis

Uji coba ini dilakukan dengan menekan tombol *print* yang ada pada menu *print* pada aplikasi.

4.5 Standar Operasional Prosedur (SOP) Proses Kerja Sistem Parkir



Gambar 4. 41 SOP Proses Kerja Sistem Parkir

BAB V

PENUTUP

5.1 Kesimpulan

Berdasarkan dari pembuatan alat “Simulasi Sistem *Smart Parking*” dapat disimpulkan bahwa:

- *Print* denah parkir pada karcis dan LCD yang menginformasikan *area* parkir yang tersedia dapat membantu pengendara mengetahui dan mencari *area* parkir yang tersedia.
- Sistem monitor dan pembayaran otomatis dapat mengefisiensi kinerja dari petugas. Ditambah sistem yang bisa menyimpan data parkir mobil sehingga petugas dapat melihat data mobil yang parkir sebelumnya.

5.2 Saran

Dari pengerjaan proyek akhir ini ada beberapa saran untuk mengembangkan proyek akhir ini kedepannya, yaitu:

- Pembaharuan pada sistem *database* sehingga *print* karcis untuk *area* parkir bisa dilakukan secara otomatis.
- Dan juga pembaharuan pada sistem pembayaran bisa dilakukan otomatis ketika mobil akan keluar.

DAFTAR PUSTAKA

- [1] T. Pribadi and P. Y. Mali, "Butuh 21 Menit Mencari Tempat Parkir di Jakarta," *VIVA Networks*, 27 Oktober 2017. [Online]. Available: www.viva.co.id/otomotif/mobil/971400-survei-butuh-21-menit-mencari-tempat-parkir-di-jakarta. [Diakses 13 Juli 2020].
- [2] G. R. Pradana, "SMART PARKING BERBASIS ARDUINO UNO," *E-Jurnal Prodi Teknik Elektronika Edisi Proyek Akhir D3*, pp. 7-8, 04 Desember 2015.
- [3] A. V. F. A. T. Inna Prastiwi H, "IMPLEMENTASI SISTEM PARKIR CERDAS DI UNIVERSITAS TELKOM. SUBSISTEM: BASISDATA DAN WEB SERVER," 2015. [Online]. Available: https://openlibrary.telkomuniversity.ac.id/pustaka/files/105596/jurnal_eproc/i-plementasi-sistem-parkir-cerdas-di-universitas-telkom-subsistem-basis-data-dan-web-server.pdf. [Diakses 15 Juli 2020].
- [4] J. Ohoiwutun, "ANALISIS DAN PERANCANGAN SMART DUMP MENGGUNAKAN ARDUINO MEGA 2560 Rev3 DAN GSM SIM900," *Jurnal Electro Luceat*, vol. 4, no. 1, p. 3, 2018.
- [5] S. M. Alimuddin, "SISTEM PARKIR CERDAS SEDERHANA BERBASIS ARDUINO MEGA 2560 Rev3," *Jurnal Electro Luceat*, vol. 4, no. 1, pp. 3-4, 2018.
- [6] L. Elektronika, "ARDUINO MEGA 2560 MIKROKONTROLER ATmega2560," 27 Februari 2017. [Online]. Available: <http://www.labelektronika.com/2017/02/arduino-mega-2560-mikrokontroler.html>. [Diakses 14 Juli 2020].
- [7] D. Wahyuni, "PERANCANGAN PROTOTYPE SMART PARKING SYSTEM SEBAGAI INFORMASI KETERSEDIAAN TEMPAT PARKIR BERBASIS ARDUINO MEGA 2560," *Skripsi*, pp. 4-6, Juli 2019.

- [8] E. Sakti, "Cara Kerja Sensor Ultrasonik, Rangkaian, & Aplikasinya," Mei 2015. [Online]. Available: <https://www.elangsakti.com/2015/05/sensor-ultrasonik.html>. [Diakses 14 Juli 2020].
- [9] A. Purnama, "LCD (Liquid Crsytal Display)," 30 Desember 2018. [Online]. Available: <https://elektronika-dasar.web.id/lcd-liquid-cristal-display/>. [Diakses 14 Juli 2020].
- [10] S. Ajie, "BEKERJA DENGAN I2C LCD DAN ARDUINO," 27 Juni 2016. [Online]. Available: <http://saptaji.com/2016/06/27/bekerja-dengan-i2c-lcd-dan-arduino/>. [Diakses 14 Juli 2020].
- [11] Components101, "SERVO MOTOR SG90 BASICS, PINOUT, WIRE DESCRIPTION, DATASHEET," 18 September 2017. [Online]. Available: <https://components101.com/servo-motor-basics-pinout-datasheet>. [Diakses 2020 Juli 2020].
- [12] S. M. Deval Gusrion, "MEMBUAT APLIKASI PENYIMPANAN DAN PENGOLAHAN DATA DENGAN VB.NET," *UPI YPTK Jurnal KomTekInfo*, vol. 5, no. 1, pp. 150-163, 2018.
- [13] A. K. Nina Oktarina, "PEMBELAJARAN BERBASIS IT APLIKASI PROGRAM MS.ACCESS UNTUK MENINGKATKAN PEMAHAMAN MAHASISWA PADA POKOK BAHASAN INVENTARISASI," *Jurnal Penelitian Pendidikan*, vol. 29, p. 161, 2011.

LAMPIRAN 1 : DAFTAR RIWAYAT HIDUP