PROYEK AKHIR – VE180626

ESTIMASI KAPASITAS DAN INDIKATOR KESEHATAN BATERAI PADA *DIVER PROPULSION VEHICLE* MENGGUNAKAN METODE COULOMB COUNTING

Muhammad Nur Aziz Asfar Afif
NRP. 10311600000018

Pembimbing
Imam Arifin, S.T., M.T.
Dr. Eng. Imam Wahyudi Farid, S.T., M.T.
Agung Imam Rahmanto, S.T.

DEPARTEMEN TEKNIK ELEKTRO OTOMASI
Fakultas Vokasi
Institut Teknologi Sepuluh Nopember
Surabaya 2019

PROYEK AKHIR – VE180626

ESTIMASI KAPASITAS DAN INDIKATOR KESEHATAN PADA BATERAI *DIVER PROPULSION VEHICLE* MENGGUNAKAN METODE COULOMB COUNTING

Muhammad Nur Aziz Asfar Afif
NRP. 10311600000018

Pembimbing
Imam Arifin, S.T., M.T.
Dr. Eng. Imam Wahyudi Farid, S.T., M.T.
Agung Imam Rahmanto, S.T.

DEPARTEMEN TEKNIK ELEKTRO OTOMASI
Fakultas Vokasi
Institut Teknologi Sepuluh Nopember
Surabaya 2019

FINAL PROJECT – VE180626

ESTIMATION OF CAPACITY AND HEALTH INDICATOR IN DIVER PROPULSION VEHICLE BATTERY USING COULOMB COUNTING METHOD

Muhammad Nur Aziz Asfar Afif
NRP. 10311600a000018

Supervisors
Imam Arifin, S.T., M.T.
Dr. Eng. Imam Wahyudi Farid, S.T., M.T.
Agung Imam Rahmanto, S.T.

DEPARTMENT OF ELECTRICAL AUTOMATION ENGINEERING
Faculty of Vocations
Institut Teknologi Sepuluh Nopember
Surabaya 2019

ESTIMASI KAPASITAS DAN INDIKATOR KESEHATAN PADA BATERAI *DIVER PROPULSION VEHICLE* MENGGUNAKAN METODE COULOMB COUNTING

PROYEK AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Memperoleh Gelar Ahli Madya Teknik
Pada
Departemen Teknik Elektro Otomasi
Fakultas Vokasi
Institut Teknologi Sepuluh Nopember

Menyetujui:

Pembimbing 1	Pembimbing 2
Imam Arifin, S.T., M.T.	Dr. Eng. Imam Wahyudi Farid, S.T., M.T.
NIP. 19730222 200212 1 001	NPP. 19902019111086

21/08/'19

SURABAYA
JULI, 2019

ESTIMASI KAPASITAS DAN INDIKATOR KESEHATAN BATERAI PADA *DIVER PROPULSION VEHICLE* MENGGUNAKAN METODE COULOMB COUNTING

Muhammad Nur Aziz Asfar Afif
10311600000018

Pembimbing I : Imam Arifin, S.T., M.T.
Pembimbing II : Dr. Eng. Imam Wahyudi Farid, S.T., M.T.
Pembimbing III : Agung Imam Rahmanto, S.T.

ABSTRAK

Kendaraan selam bawah air atau *Diver Propulsion Vehicle* (DPV) menggunakan baterai sebagai penyimpan energi listrik. Di dalam penggunaanya, baterai dapat mengalami penurunan kapasitas. Salah satu faktor yang dapat mempercepat penurunan kapasitas baterai adalah kondisi *overcharging* dan *overdischarging*. Untuk mengatasi permasalahan tersebut, dilakukan perancangan estimasi kapasitas dan indikator kesehatan pada baterai. Perancangan tersebut digunakan untuk mengetahui kondisi baterai ketika dipakai oleh pengguna. Metode yang digunakan adalah melakukan estimasi kapasitas dengan parameter SOC dan indikator kesehatan baterai dengan parameter SOH. Nilai SOC didapatkan dengan menjumlahkan arus yang terukur oleh sensor. Sedangkan nilai SOH, didapatkan dengan membandingkan nilai arus total yang terukur dengan nilai desain kapasitas pabrik. Nilai kapasitas dan kesehatan disimpan pada modul SD *Card*.

Berdasarkan pengujian yang dilakukan, didapatkan nilai rata-rata eror (RMSE) pada sensor arus ACS 758 sebesar 0.02273, selisih ketika kondisi *discharging* sebesar 3.1%, dan pada kondisi *charging* sebesar 4.5% dari penghitungan kapasitas secara manual. Selain itu, didapatkan nilai indikator kesehatan baterai sebesar 93.89%.

Kata kunci : SOC, SOH, *Data Logging*

Halaman ini sengaja dikosongkan

ESTIMATION OF CAPACITY AND HEALTH INDICATOR IN DIVER PROPULSION VEHICLE BATTERY USING COULOMB COUNTING METHOD

Muhammad Nur Aziz Asfar Afif
10311600000018

Supervisor I : Imam Arifin, S.T., M.T.
Supervisor II : Dr. Eng. Imam Wahyudi Farid, S.T., M.T.
Supervisor III : Agung Imam Rahmanto, S.T.

ABSTRACT

Under water vehicle or Diver Propulsion Vehicle (DPV) uses battery as energy storage. In battery, the capacity can be decreased . One of the factor that can make fasten battery capacity decreased is the condition of overcurrent charging and overdischarging. One method that can used to monitor the battery condition when battery used by user is with adding parameter State Of Charge (SOC) for estimating capacity of battery an State Of Health (SOH) for estimating battery health in general also data logging as data saver. The value of SOC parameter is gotten by summing all data that measured by sensor and compared by nominal capacity which designed by factory. The value of battery capacity and health is saved in SD Card Module.

Based on testing current sensor ACS 758 had average eror (RMSE) 0.02273 and the testing is continued by battery DPV when discharging condition obtained the difference 3.1% and when battery in charging condition obtained difference 4.5% from manual counting. Besides from the battery test, health indicator in battery is gotten. The battery health is 93.89%.

Keyword : *SOC, SOH, Data logging*

Halaman ini sengaja dikosongkan

KATA PENGANTAR

Puji syukur penulis panjatkan kepada Allah SWT karena atas rahmat dan karuniaNya laporan Proyek Akhir dengan judul "Estimasi Kapasitas dan Indikator Kesehatan pada Baterai *Diver Propulsion Vehicle* menggunakan Metode Coulomb Counting*"* dapat terselesaikan sesuai waktu yang telah ditentukan. Proyek Akhir merupakan salah satu syarat kelulusan mahasiswa di Departemen Teknik Elektro Otomasi ITS.

Dalam menyelesaikan laporan ini, penulis dibantu oleh berbagai pihak. Oleh karena itu, pada kesempatan ini penulis menyampaikan rasa terimakasih kepada keluarga, khususnya orang tua yang senantiasa memberikan dukungan dan doa. Bapak Imam Arifin, S.T., M.T., selaku kepala Laboratorium Cyphiral, dosen wali, dan pembimbing I. Bapak Dr. Eng. Imam Wahyudi Farid, S.T., M.T., selaku pembimbing II, atas bantuan dan bimbingannya dalam pengerjaan Proyek Akhir. Bapak Agung Imam Rahmanto, S.T., selaku pembimbing III dari perusahaan yang telah memberikan bimbingan selama di perusahaan dari awal hingga akhir pengerjaan. Seluruh karyawan PT Bhimasena Research and Development, khususnya pada divisi *under water* karena telah banyak membantu dalam proses pengerjaan. Segenap Direksi PT Bhimasena Research and Development yang telah memberikan izin untuk melakukan peneltian di tempat tersebut. Serta seluruh pihak yang tidak bisa disebutkan satu persatu, atas bantuan dan dukungan, sehingga penulis dapat menyelesaikan pengerjaan Proyek Akhir ini.

Penulis menyadari bahwa dalam penyusunan laporan ini tidak terlepas dari kekurangan, oleh karena itu penulis mengharapkan kritik dan saran yang membangun dari pembaca. Semoga laporan ini dapat memberikan manfaat bagi penulis khususnya dan pembaca pada umumnya.

Surabaya, Juli 2019

Muhammad Nur Aziz Asfar Afif
NRP. 10311600000018

Halaman ini sengaja dikosongkan

DAFTAR ISI

Halaman ini sengaja dikosongkan

DAFTAR GAMBAR

Halaman ini sengaja dikosongkan

DAFTAR TABEL

Halaman ini sengaja dikosongkan

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Indonesia merupakan suatu negara maritim dengan wilayah perairan yang luas. Untuk menjaga kedaulatan wilayah tersebut, diperlukan teknologi guna menunjang kebutuhan militer. Salah satu perusahaan yang mengembangkan dan memproduksi teknologi untuk kebutuhan militer adalah PT. Bhimasena Research and Development. Perusahaan tersebut mempunyai beberapa produk, salah satu produk yang dihasilkan adalah kendaraan selam bawah air atau yang sering disebut dengan *Diver Propulsion Vehicle* (DPV). Kendaraan ini berfungsi untuk membantu pergerakan personil ketika melakukan penyelaman di bawah air [1].

Untuk melakukan kegiatan operasional tersebut, DPV dilengkapi dengan baterai lithium iron phosphate ($LiFePO_4$) sebagai penyimpan energi listrik [1]. Baterai pada DPV memiliki tegangan maksimal sebesar 54 volt, baterai tersebut terdiri dari beberapa sel baterai yang dihubungkan secara seri dan paralel.

Pada penggunaanya, kapasitas baterai dapat mengalami penurunan. Beberapa faktor yang dapat mempercepat penurunan kapasitas pada baterai adalah kondisi *overcharging* maupun kondisi *overdischarging* [2]. Baterai DPV sudah mempunyai *Battery Management System* (BMS), namun BMS tersebut belum dapat melakukan estimasi kapasitas dan kesehatan baterai. Oleh karena itu, dilakukan perancangan untuk mengetahui kondisi baterai pada ketika dipakai oleh pengguna. Metode yang digunakan adalah melakukan estimasi kapasitas dengan parameter *State Of Charge* (SOC) dan kesehatan baterai dengan parameter *State Of Health* (SOH). Selain itu, untuk menyimpan data yang dihasilkan ditambahkan *data logging* sebagai sistem penyimpanan data baterai pada DPV.

Melalui penelitian ini, diperoleh informasi utama berupa estimasi kapasitas dan kesehatan baterai. Selain itu, terdapat informasi tambahan berupa arus masuk, arus keluar, nilai tegangan baterai.

1.2 Rumusan Masalah

BMS yang digunakan pada baterai DPV memiliki beberapa keterbatasan. Perangkat tersebut hanya bisa memotong arus (*cut off*) baterai yang terhubung dengan beban ketika baterai berada diluar batas kapasitasnya. BMS tersebut belum dapat melakukan estimasi kapasitas baterai, kesehatan baterai, dan penyimpanan data hasil *monitoring* baterai yang dapat menggambarkan pemakaian baterai pada saat DPV dipakai oleh pengguna.

1.3 Batasan Masalah

Dalam melakukan penelitian ini, terdapat beberapa hal yang menjadi batasan, yaitu :

1. Validasi yang dilakukan pada sensor arus ACS 758 mempunyai nilai maksimum 3 A.
2. Data yang diambil dari pengujian baterai adalah tegangan, arus masuk, dan arus keluar.
3. Penelitian difokuskan pada kapasitas dan kesehatan baterai.

1.4 Tujuan

Dari perancangan sistem ini, *board monitoring* baterai dapat memberikan informasi mengenai estimasi kapasitas baterai, kesehatan baterai, serta penyimpanan data hasil *monitoring* baterai yang digunakan untuk menggambarkan kondisi baterai ketika baterai dipakai oleh pengguna. Beberapa parameter yang didapatkan diharapkan dapat dijadikan sebagai data perekaman, data perekaman tersebut digunakan perusahaan ketika perawatan dan penganalisaan kerusakaan baik pada baterai maupun pada DPV secara keseluruhannya.

1.5 Metodologi

Pada penelitian ini terdapat beberapa tahapan yang dilakukan, adapun tahapan yang dilakukan adalah sebagai berikut :

1. Studi Pustaka dan Survei Literatur
   Studi pustaka dan survei literatur dilakukan dengan pengumpulan materi yang berkaitan dengan topik penelitian yang bersumber dari buku pustaka, laporan, paper jurnal, paper konferensi, dan *manual*

book atau *reference book* dari perangkat yang digunakan. Materi yang diperoleh kemudian dipelajari dan dijadikan sebagai acuan dalam diskusi dengan pembimbing, perancangan perangkat keras, perangkat lunak, pengujian, dan analisis.

2. Perancangan Sistem
   Materi yang didapatkan kemudian dipelajari dan digunakan pada saat proses perancangan eksperimen, perangkat keras, perangkat lunak, dan analisa dalam pengerjaan proyek akhir mengenai Estimasi Kapasitas dan Indikator Kesehatan pada Baterai *Diver Propulsion Vehicle* Menggunakan Metode Coulomb Counting.
3. Pengujian dan Analisa Data
   Pengujian dilakukan untuk mengetahui kinerja metode yang digunakan dan menguji kemampuan alat yang telah dibuat. Hasil pengujian meliputi nilai kapasitas baterai, kesehatan baterai, karakterisitik baterai akan dibandingkan terhadap teori penunjang dan perangkat pembanding guna mengurangi tingkat kesalahan dari hasil pengujian.
4. Penyusunan Laporan
   Penyusunan laporan dilakukan berdasarkan proyek akhir yang yang telah dikerjakan. Laporan yang telah selesai disusun akan dibuat dalam bentuk *soft file* dan *hard file* sebagai arsip laporan proyek akhir.

1.6 Sistematika Penulisan

Penelitian yang dilakukan disusun di dalam laporan dengan pembagian sebanyak lima bab, dengan susunan sebagai berikut :

BAB I PENDAHULUAN
Berisi tentang latar belakang, rumusan masalah, batasan masalah, tujuan, metodologi, dan sistematika penulisan laporan buku Proyek Akhir (PA).

BAB II PENUNJANG KESEHATAN DAN INDIKATOR BATERAI
Menjelaskan mengenai teori-teori penunjang yang digunakan sebagai landasan dalam melakukan penelitian.

BAB III ESTIMASI KAPASITAS DAN INDIKATOR KESEHATAN

Menjelaskan langkah - langkah yang dilakukan dalam melakukan perancangan eksperimen, perangkat lunak, dan perangkat keras pada pembuatan sistem.

BAB IV PENGUJIAN DAN ANALISIS

Membahas mengenai hasil data pengujian dan analisis data yang diperoleh.

BAB V PENUTUP

Pada bagian ini berisi mengenai kesimpulan yang didapatkan dan saran untuk penelitian selanjutnya.

BAB 2
PENUNJANG KESEHATAN DAN INDIKATOR BATERAI

2.1 *Diver Propulsion Vehicle* (DPV)

DPV merupakan suatu kendaraan bawah air yang digunakan oleh penyelam SCUBA dan *rebreather* bawah air untuk meningkatkan jangkauan jelajah dan kecepatan ketika melakukan penyelaman. Pada Gambar 2.1, merupakan bentuk dari DPV yang didesain untuk memenuhi kebutuhan mekanikal, elektrikal, dan sistem propulsi. DPV dilengkapi dengan baterai lithium iron phosphate ($LiFePO_4$) sebagai penyimpan energi listrik dan sebuah sistem informasi yang digunakan untuk memudahkan pengoperasian kendaraan dimana sistem tersebut juga berfungsi untuk menjaga keamanan penyelam serta kendaraan itu sendiri. Informasi yang ditampilkan berupa waktu jelajah, sistem navigasi, indikator baterai, dan kecepatan jelajah kendaraan yang ditampilkan pada sebuah layar LCD [1].

Gambar 2.1 *Diver Propulsion Vehicle* (DPV) [1]

2.2 Baterai Lithium Iron Phosphate ($LiFePO_4$)

Baterai $LiFePO_4$ merupakan sebuah baterai ramah lingkungan yang tidak mempunyai kandungan zat berbahaya. Baterai berjenis lithium iron phosphate ($LiFePO_4$) dikembangkan sebagai solusi keandalan dan keterbatasan pada kelangkaan logam (Ni, Co, Mn, dll) yang digunakan baterai pada umumnya. Baterai $LiFePO_4$ menggunakan

phosphate sebagai material katoda, struktur katoda dari baterai ini memberikan reaksi kimia yang stabil dan mempunyai ketahanan panas yang menjadikan baterai $LiFePO_4$ sebagai baterai berjenis lithium yang paling aman digunakan. Disisi lain, baterai $LiFePO_4$ mempunyai tingkat potensi listrik yang rendah yaitu (3.2 – 3.3 V) [3][4]. Bentuk dari baterai $LiFePO_4$ yang digunakan pada DPV terdapat pada Gambar 2.2.

Gambar 2.2 Baterai $LiFePO_4$ DPV

Battery Model	**48V 40AH**
Nominal Capacity (AH)	40Ah
Nominal Voltage (V)	51.2
Source Resistance (mΩ)	<40
Cell Combination	16-Series, 2-Paralles
Cell Type	Rectangle Cell (3.2V 20Ah)
Cell Quantity (series*parallels)	32 pcs
Cell Size	198x140x10mm
Battery Size (**without BMS**)	325x208x148mm or 295×208×165mm
Battery Size (**with BMS Box**)	325x208x183mm or 295×208×200mm
Discharge Cutoff Voltage (V)	38.4
Charge Cutoff Voltage (V)	58.4
Rated Discharge Current (A)	40
Maximum Continuous Discharge Current (A)	80
Instantaneous Maximum Discharge Current (A)	160
Maximum Continuous Charge Current (A)	10
Maximum Continuous BMS Limited Current (A)	40
Maximum Instantaneous BMS Limited Current (A)	80
Charge Mode	CC-CV
Standard Charge Current (A)	2
Charge Time under Standard Charge Current	22.5 hours
Fast Charge Current (A)	5
Charge Time under Fast Charge Current	9 hours
Charge Temperature Range	0-55℃
Discharge Temperature Range	-10-60℃
Battery Net Weight (include BMS) (kg)	17.5
Battery Gross Weight (include Package) (kg)	20
Charging Cycles	>1000 times
Recommend Application	Less than 1800W Motor

Gambar 2.3 Spesifikasi Baterai $LiFePO_4$ DPV

2.3 Baterai Lithium Ion

Baterai lithium ion (Li-Ion) merupakan tipe baterai yang paling modern dan digunakan sistem kendaraan listrik sebagai sistem tenaga saat ini. Keuntungan dari baterai lithium adalah mempunyai siklus hidup yang panjang, rentang suhu yang luas, memiliki tingkat *selfdischarge* rendah, kemampuan pengisian yang cepat, mempunyai efisiensi dan tingkat kepadatan energi yang tinggi. Di samping itu, baterai tersebut mempunyai kekurangan dimana baterai tersebut tidak mempunyai kemampuan untuk mendapatkan nilai arus yang lebih tinggi dari nilai arus maksimum, ketidaktahanan terhadap suhu yang tinggi dan dapat mengakibatkan adanya ledakan [5].

Capacity(2)	Minimum	3350mAh
	Typical	3450mAh
Nominal voltage		3.6V
Charging	Method	CC-CV
	Voltage	4.20V
	Current	Std. 1475mA
	Time	Std. 270 min.
Weight (max.)		48.0g
Temperature	Charge	10 to +45° C
	Discharge	-20 to +60° C
	Storage	-20 to +50° C
Energy density(3)	Volumetric	693 Wh/l
	Gravimetric	224 Wh/kg

Gambar 2.4 Spesifikasi Baterai Lithium Ion (Li-Ion)

2.4 *State Of Charge* (SOC)

SOC merupakan sebuah parameter yang memberikan informasi mengenai kapasitas yang tersisa pada baterai, nilai SOC dinyatakan dalam satuan presentase. SOC merupakan salah satu parameter penting yang diperlukan untuk memastikan pengisian arus dan pengeluaran arus yang aman [2]. Informasi yang diberikan oleh SOC dapat membantu memberikan keputusan yang tepat pada proses pengisian dan

pengosongan arus untuk menjaga kerusakan baterai dari kondisi kelebihan pengisian arus dan pengosongan arus. Nilai dari SOC berubah berdasarkan arus yang masuk dan keluar dari sel baterai, nilai tersebut dapat diperkirakan menggunakan metode Coulomb Counting [6].

2.4.1 Coulomb Counting

Coulomb Counting merupakan suatu metode yang digunakan untuk mengukur kapasitas baterai (SOC) dengan menghitung nilai arus yang masuk atau keluar dari baterai, nilai tersebut dijumlahkan terhadap waktu. Nilai dari pengukuran SOC harus dikurangi dengan kapasitas baterai sebelumnya [7]. Nilai SOC dapat diperoleh menggunakan persamaan 2.1.

$$SOC(t) = SOC_0 - \frac{100}{3600 \text{ x } Cn} \int_0^t i(\tau) d\,\tau \qquad (2.1)$$

Pada persamaan (1) SOC(t) merupakan nilai kapasitas saat ini, SOC_0 adalah nilai SOC pada saat kondisi kapasitas baterai penuh 100% atau pada saat baterai kosong 0%. I(τ) d τ merupakan total arus yang ditarik atau masuk pada baterai. Sedangkan Cn adalah kapasitas yang tertera pada label baterai atau kapasitas nominal baterai [8].

2.5 *State Of Health* (SOH)

SOH merupakan sebuah parameter yang memberikan informasi mengenai berkurangnya performa pada baterai [6]. Nilai SOH memberikan estimasi mengenai kondisi kesehatan baterai secara umum. Kapasitas maksimal dari baterai (dinyatakan dalam satuan mAh) berkurang setiap kali baterai digunakan dan dilakukan pengisian ulang. SOH mendefinisikan kapasitas baterai yang sebenarnya dari kondisi awal baterai [7], SOH didefinisikan menggunakan persamaan 2.

$$\text{SOH} = \frac{Ccap}{Cnominal} \times 100\% \qquad (2.2)$$

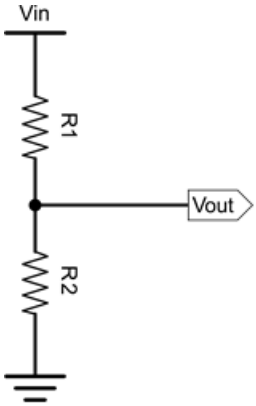
Pada persamaan 2.2, C_{cap} merupakan kapasitas baterai yang sebenarnya dan $C_{nominal}$ merupakan kapasitas yang terdapat pada label baterai. Pada kondisi kapasitas baterai dibawah 80% dari kapasitas awal, BMS akan memberikan sinyal peringatan yang menunjukkan bahwa baterai harus

diganti. Dengan mengetahui nilai estimasi parameter SOH, keputusan yang tepat untuk penggantian baterai dengan performa yang kurang baik pada kendaraan listrik dengan baterai baru dapat dilakukan [6].

2.6 *Battery Management System* (BMS)

Baterai merupakan salah satu bagian utama pada kendaraan listrik, *Battery Management System* (BMS) digunakan untuk menjaga baterai bekerja dalam kondisi optimal dan memperpanjang masa pakai [6]. Sebuah perangkat BMS melakukan pengecekan dan melaporkan keseluruhan informasi mengenai baterai kepada pengguna, BMS juga dapat melakukan beberapa hal diantaranya *monitoring* tegangan setiap sel baterai, arus, temperatur, kapasitas baterai, kesehatan baterai, menyeimbangkan tegangan tiap sel baterai, dan memotong arus yang terhubung pada baterai ketika terjadi kondisi kelebihan pengisian serta pengeluaran arus [7].

2.7 Rangkaian Pembagi Tegangan



Gambar 2.5 Rangkaian Pembagi Tegangan

Gambar 2.5 merupakan rangkaian pembagi tegangan, rangkaian tersebut terdiri dari komponen elektronika pasif yang dapat mengubah nilai tegangan dc ke nilai tegangan dc yang lebih rendah. Pembagi tegangan terdiri dari dua resistor yang disusun secara seri dengan tegangan masukan. Tegangan masukan (Vin) melewati resistor R1 dan

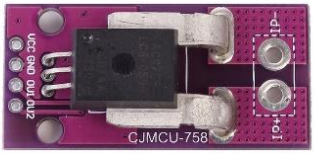
resistor R2, tegangan keluaran (Vout) merupakan penurunan tegangan pada R2. Nilai dari tegangan keluaran lebih kecil dari pada tegangan masukan (Vin) karena tegangan total melewati resistor R1 dan R2 [9]. Untuk menentukan besaran resistor yang digunakan untuk mendapatkan tegangan keluaran (Vout) pada rangkaian pembagi tegangan dapat menggunakan persamaan 2.3.

$$Vout = Vin \frac{R2}{R1+R2} \quad (2.3)$$

Pada persamaan 2.3, Vout merupakan keluaran tegangan setelah melalui pembagi tegangan, Vin merupakan nilai tegangan masukan yang akan diturunkan nilainya. Sedangkan R1 dan R2 merupakan resistor yang digunakan pada rangkaian pembagi tegangan.

2.8 ACS 758

ACS 758 merupakan sebuah modul sensor yang digunakan untuk mendeteksi besar arus, arus yang mengalir melewati blok terminal dari modul sensor akan dideteksi menggunakan IC ACS 758 LCB – 050B yang memanfaatkan efek hall. Efek hall merupakan hasil dari perbedaan tegangan pada hall yang melintasi konduktor listrik. Modul sensor ini mempunyai prinsip kerja dimana arus yang mengalir melalui tembaga menghasilkan medan magnet yang diubah IC Hall menjadi tegangan yang proporsional [10]. Sensor ini digunakan untuk mengetahui arus yang masuk maupun arus yang keluar pada baterai. Bentuk dari sensor arus ACS 758 dapat dilihat pada Gambar 2.6.

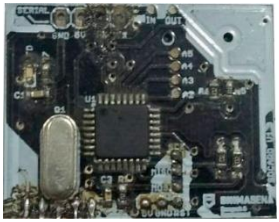


Gambar 2.6 Sensor Arus ACS 758

2.9 Modul SD *Card*

Modul SD *Card* merupakan modul *data logger* yang digunakan untuk menyimpan data baterai dalam bentuk file excel, modul SD *Card*

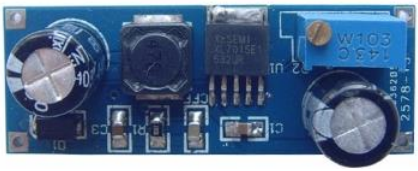
tersebut terdiri dari mikrokontroler berbasis ATmega 328P, SD *Card* slot sebagai tempat disematkannya kartu SD *Card* dan regulator ams 1117 sebagai penurun tegangan, tegangan digunakan sebagai operasi pada SD/MMC. Modul tersebut menggunakan komunikasi serial dan dibantu oleh perangkat usb to ttl untuk komunikasi dengan PC. Selain itu untuk komunikasi antara SD *Card* slot dengan ATmega 328, modul tersebut menggunakan komunikasi SPI [11]. Bentuk dari Modul SD *Card* dapat dilihat pada Gambar 2.7.



Gambar 2.7 Modul SD *Card*

2.10 XL 7015

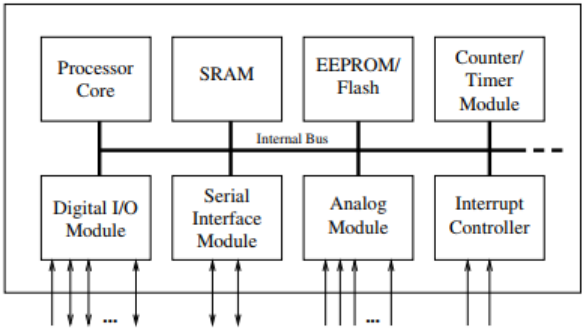
Perangkat ini merupakan modul dc to dc *buck converter* atau modul penurun tegangan dc ke dc. Modul *buck converter* mempunyai efisiensi yang cukup tinggi, modul tersebut digunakan sebagai pengganti regulator tegangan pada keluaran tegangan yang relatif tinggi. Penggunaan regulator tegangan membuang daya cukup tinggi dan memerlukan pendingin (*heatsink*) sehingga membuat ukuran regulator menjadi besar. Secara umum, *buck converter dc to dc* terdiri dari sebuah kontroler PWM, transistor, induktor, kapasitor dan diode [12]. Modul XL 7015 mempunyai operasi tegangan masukan dari 5V hingga 80V dengan keluaran yang dapat diatur dari 1,25 V hingga 20V [13]. Bentuk dari modul buck converter XL 7015 dapat dilihat pada Gambar 2.8.



Gambar 2.8 Buck Converter XL 7015

2.11 Mikrokontroler

Mikrokontroler merupakan sebuah komputer yang terdapat pada suatu rangkaian yang terintegrasi di dalam sebuah chip, mikro mempunyai arti kecil dan kontroler dapat diidentifikasi sebagai perangkat lunak untuk mengontrol suatu objek, proses, atau peristiwa. Mikrokontroler biasanya terdapat pada segala perangkat kehidupan yang digunakan untuk mengukur, menyimpan, mengontrol, menghitung, atau menampilkan informasi untuk disimpan di dalam mikrokontroler [14]. Sebuah mikrokontroler dilengkapi dengan memori, timer, I/O pin, dan peripheral lainnya. Desain dasar internal antar mikrokontroler mempunyai kemiripan yang hampir sama. Gambar 2.9 menunjukkan blok diagram mikrokontroler. Keseluruhan komponen terhubung dengan sebuah *internal bus* yang terintegrasi pada sebuah chip dan terhubung ke luar melalui pin I/O [15]. Blok diagram mikrokontroler dapat dilihat pada Gambar 2.9.



Gambar 2.9 Diagram Blok Mikrokontroler [15]

2.11.1 ATmega 328 P

ATmega 328P adalah mikrokontroler keluaran atmel yang berdaya rendah dan mempunyai arsitektur RISC (*Reduce Instruction Set Computer*). Inti ATmega menggabungkan instruksi set dengan 32 register yang terhubung langsung dengan dengan *Arithmetic Logic Unit* (ALU), yang memungkinkan dua register independen diakses dalam satu instruksi yang dijalankan dalam satu siklus *clock* [16].

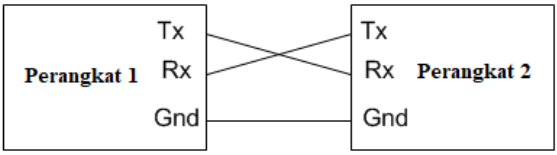
Gambar 2.10 ATmega 328P

2.11.2 Arduino IDE

Arduino IDE (*Integrated Development Environment*) merupakan sebuah perangkat lunak yang diperkenalkan oleh Arduino [17], perangkat ini berisi editor teks untuk menuliskan kode, area pesan, konsol teks, *toolbar* dengan tombol untuk fungsi umum dan serangkaian menu yang terhubung dengan perangkat keras Arduino untuk mengunggah program dan komunikasi. Arduino IDE dilengkapi dengan *serial monitor* yang terdapat pada tombol bagian kanan atas. *Serial monitor* membantu untuk melakukan *debug* dan mengetahui operasi dari program yang telah dibuat. Penggunaan *serial monitor* mengharuskan untuk menentukan kecepatan pengiriman data atau biasa yang disebut dengan *baud rate* [18].

2.11.3 Komunikasi Serial

Adalah teknik komunikasi yang digunakan pada telekomunikasi dimana transfer data terjadi dengan mentransmisikan satu bit pada satu waktu secara bergantian melalui sebuah bus komputer atau sebuah saluran komunikasi. Komunikasi serial merupakan komunikasi *full duplex* atau pengiriman dan penerimaan sinyal dapat dilakukan secara bersamaan [18].



Gambar 2.11 Komunikasi Serial

2.12 Digital Low Pass Filter

Low-pass filter merupakan suatu perhitungan yang digunakan untuk mengurangi *noise* pada suatu frekuensi atau sinyal. Filter ini bekerja dengan memotong frekuensi bernilai tinggi dan melewatkan frekuensi bernilai rendah [19], Perhitungan dari *Low-pass filter* mempunyai persamaan 2.4.

$$\mathrm{St} = \alpha \times \mathrm{Yt} + (1 - \alpha) \times \mathrm{St}_{-1} \quad (2.4)$$

Pada persamaan 2.4, St adalah hasil filter ketika waktu *t*. Yt adalah data bernilai awal. α adalah koefisien yang menentukan jumlah sampel yang dihitung dengan rentang nilai 0-1. Koefisien dengan nilai kecil mengakibatkan perhitungan yang lambat karena jumlah sampel yang banyak, namun hasil filter menjadi lebih baik, dan sebaliknya.

2.13 Root Mean Square Error (RMSE)

Root Mean Square Error (RMSE) merupakan suatu perhitungan yang digunakan untuk mengukur perbedaan antara nilai yang diprediksi melalui model hipotesis dan nilai yang diamati. Nilai RMSE dapat dihitung menggunakan persamaan 2.5.

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{n} (Yi - \hat{\mathrm{Y}}\mathrm{i})^2}{n}} \quad (2.5)$$

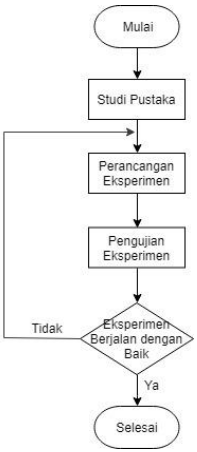
Pada persamaan 2.5 Yi merupakan data sebenarnya atau data yang telah tervalidasi, Ŷi merupakan data hasil estimasi, dan n merupakan banyak data.

BAB 3
ESTIMASI KAPASITAS DAN INDIKATOR KESEHATAN

Pada bab ini terdapat tiga sub bab yang menjelaskan mengenai perancangan eksperimen, perangkat keras, dan perangkat lunak. Penjelasan pada bab ini diawali dengan perancangan metode pada baterai, penjelasan dilanjutkan dengan perancangan akuisisi data sensor serta peralatan yang digunakan. Setelah itu penjelasan dilanjutkan dengan pembahasan mengenai langkah-langkah pembuatan perangkat keras *board monitoring* baterai dan konfigurasi pin pada perangkat yang digunakan. Terakhir, pada bab ini membahas mengenai pembuatan perangkat lunak pada *board monitoring* baterai dan modul SD *Card*.

3.1 Perancangan Eksperimen

Pada perancangan eksperimen ini menjelaskan mengenai akuisisi data dan validasi sensor arus. Selain itu, perancangan metode yang digunakan untuk mengestimasi parameter SOC dan SOH pada baterai berdasarkan landasan teori yang dijelaskan pada Bab 2. Alur yang dilakukan pada perancangan ekperimen ditunjukkan pada Gambar 3.1.

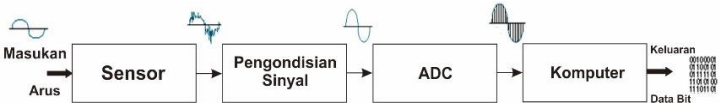


Gambar 3.1 Diagram Alir Perancangan Eksperimen

Langkah pertama yang dilakukan pada perancangan eksperimen ini adalah melakukan studi pustaka mengenai karakteristik baterai dan sensor arus. Selanjutnya, perancangan eksperimen dilanjutkan dengan perancangan metode yang akan dilakukan. Pada perancangan eksperimen ini terdiri dari dua tahapan metode, tahapan yang pertama yaitu melakukan akusisi data dan validasi sensor arus kemudian dilanjutkan pembandingan arus masuk dan arus keluar serta pembandingan antara nilai kapasitas baterai yang terukur dengan nilai kapasitas baterai yang tertera pada *datasheet* baterai.

3.1.1 Akuisisi Data dan Validasi Sensor Arus

Tahap pertama pada penelitian ini adalah melakukan akuisisi data dan validasi sensor arus ACS 758. Pada proses akuisisi data dan pengubahan sinyal analog menjadi bentuk digital terdapat beberapa persyaratan yang harus dipenuhi, syarat untuk terjadinya akuisisi data terdapat pada Gambar 3.2.

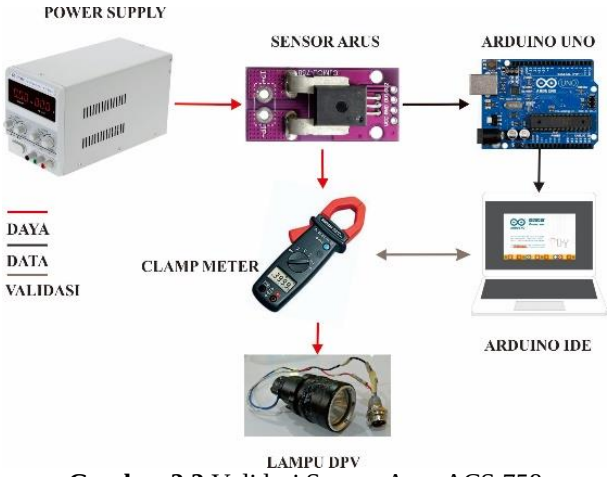


Gambar 3.2 Akuisisi Data Sensor Arus ACS 758

Pada Gambar 3.2 akuisisi data dimulai dengan pengukuran fenomena fisik atau besaran fisis oleh sensor, masukan dari sensor tersebut berupa sinyal analog.8 Data yang diperoleh sensor terlebih dahulu melalui proses pengondisian sinyal, proses pengondisian sinyal dilakukan agar sinyal yang dihasilkan oleh sensor dapat diproses oleh mikrokontroler serta untuk mengurangi *noise* yang terdapat pada sensor. Setelah itu sinyal analog dikonversikan menjadi digital menggunakan ADC mikrokontroler yang menghasilkan data dengan bentuk kode bit.

Setelah itu, perancangan eksperimen dilanjutkan dengan validasi sensor arus. Validasi dilakukan untuk mengetahui tingkat keakuratan pembacaan sensor. Pada validasi ini, *power supply* digunakan untuk memberikan daya pada beban lampu. Untuk melakukan validasi, diberikan nilai arus yang bervariasi pada lampu. Pada tahap validasi ini, sensor arus diletakkan secara seri antara dc *power supply* dan lampu. Untuk melakukan validasi, digunakan nilai keluaran arus dari clamp

meter sebagai data pembanding sensor arus. Data yang dihasilkan oleh sensor arus ditampilkan pada Arduino IDE.



Gambar 3.3 Validasi Sensor Arus ACS 758

3.1.2 Perekaman Data 1 Sel Baterai

Sebelum melakukan pengujian pada baterai DPV, dilakukan perancangan metode terhadap baterai 1 sel dan validasi sensor arus terlebih dahulu. Validasi sensor arus dilakukan dengan memberikan beban dengan nilai arus yang berbeda – beda. Setelah data yang dihasilkan oleh sensor arus telah mendekati alat ukur pembanding, yaitu clamp meter. Perancangan eksperimen dilanjutkan dengan melakukan *static test* pada 1 sel baterai.

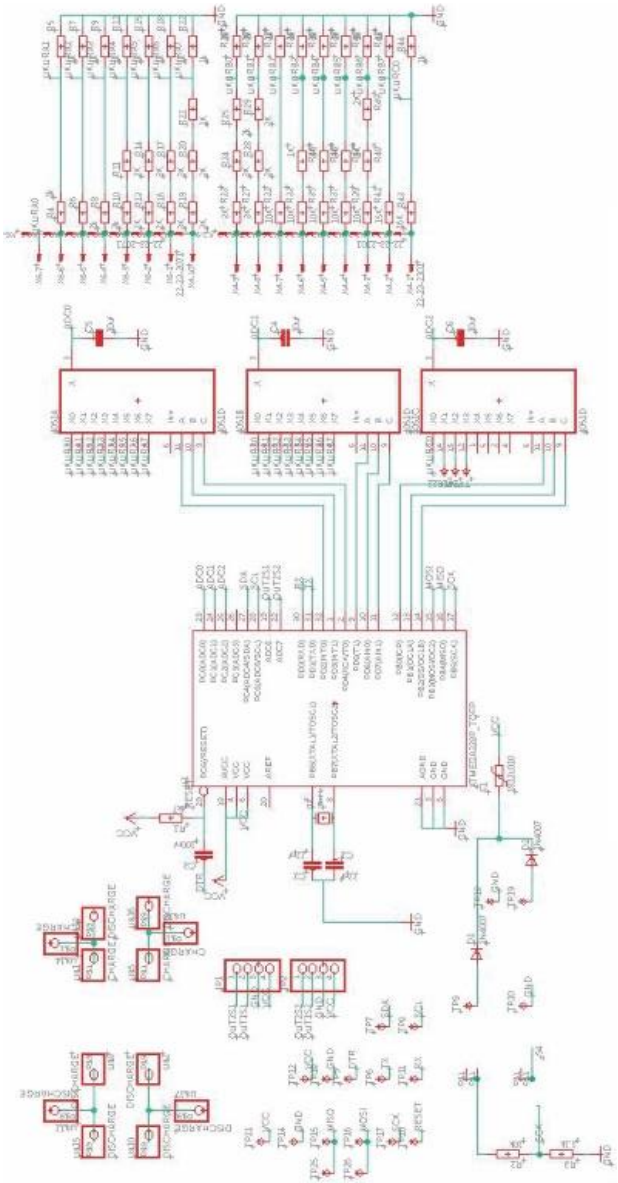
Pengujian *static test* pada 1 sel baterai menggunakan baterai bejenis lithium-ion, baterai tersebut mempunyai tegangan maksimum 4.2 V dengan kapasitas 3450 mAh. Pengujian pada baterai lithium-ion 1 sel tersebut dilakukan dengan melakukan pengisian terlebih dahulu hingga baterai mencapai kondisi penuh (SOC = 100%) menggunakan modul *charger* 1 sel baterai. Setelah itu, baterai yang telah terisi penuh dilakukan pengosongan arus dengan beban lampu DPV. Pengosongan arus dilakukan hingga baterai mencapai batas bawah nilai tegangan baterai. Pengujian ini dilakukan untuk mengetahui kapasitas (mAh) baterai sesungguhnya.

3.2 Perancangan Perangkat Keras

Pada tahap perancangan perangkat keras ini menjelaskan mengenai pembuatan *board monitoring* baterai. Pada *board monitoring* merupakan tempat disematkannya sensor Arus ACS 758. Selain itu terdapat konfigurasi pin sensor arus dan SD *Card* yang digunakan untuk melakukan *logging* data.

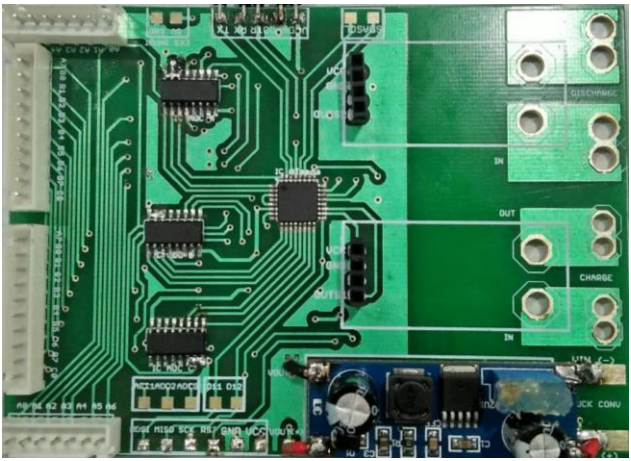
3.2.1 Perancangan *Board Monitoring* Baterai

Dalam melakukan penelitian ini dirancang sebuah *board monitoring* baterai, *board* tersebut terdiri dari mikrokontroler berbasis ATmega 328P, *crystal oscillator* (XTAL) 16 MHz, serta beberapa komponen elektronika pendukung seperti dioda, resistor, dan kapasitor. Selain itu, karena *board monitoring* baterai juga mengambil daya dari baterai maka untuk memberikan proteksi apabila terjadi arus yang berlebih digunakan *fuse* sebesar 500 mA yang ditempatkan pada keluaran dari buck converter. *Board monitoring* merupakan tempat disematkan modul sensor arus ACS 758 dan buck converter XL 7015. *Board* ini berfungsi sebagai tempat pengolahan data lama waktu pengecasan, berapa kali pengecasan, nilai SOC, dan SOH sebelum dikirimkan pada modul SD *Card*. Gambar 3.4 merupakan bentuk skematik dari *board monitoring* baterai.



Gambar 3.4 Skematik *Board Monitoring* Baterai

Pada pembuatan *board monitoring* baterai, terdapat beberapa tahapan yaitu perancangan rangkaian, pembuatan skematik, penyablonan PCB, dan penyolderan komponen. Hasil dari pembuatan *board monitoring* terdapat pada Gambar 3.5.



Gambar 3.5 Hasil Pembuatan *Board Monitoring* Baterai

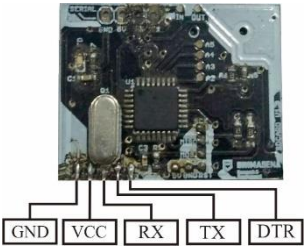
3.2.2 Konfigurasi ACS 758

Untuk melakukan pengukuran arus, sensor arus ACS 758 diletakkan pada *board monitoring* baterai. Rangkaian dari baterai dan beban terhubung pada bagian 1 dan 2 yang digunakan untuk mendeteksi arus. Pin VCC pada modul tersebut terhubung dengan tegangan keluaran 5 V dari *buck converter*. Pin GND pada sensor terhubung dengan pin GND dari *buck converter*. Sedangkan pin OUT 2 dari sensor yang digunakan sebagai keluaran sensor terhubung dengan pin ADC mikrokontroler. Bentuk dari sensor arus ACS 758 dapat dilihat pada Gambar 3.6.

Gambar 3.6 Konfigurasi ACS 758

3.2.3 Konfigurasi Modul SD *Card*

Modul SD *Card* merupakan sebuah modul yang terdiri dari mikrokontroler berbasis ATmega 328P, SD *Card* slot, *crystal oscillator* 16 Mhz, *voltage regulator* AMS 1117, serta beberapa komponen elektronika pendukung lainnya seperti resistor dan kapasitor. *Board* tersebut berfungsi sebagai tempat dikirimkannya data dari *board monitoring* baterai. *Board* tersebut menggunakan usb to ttl untuk melakukan komunikasi dengan PC.



Gambar 3.7 Konfigurasi Modul SD *Card*

3.3 Perancangan Perangkat Lunak

Pada tahap perancangan perangkat keras, terdiri dari 2 tahapan pemrograman yaitu pemrograman pada *board monitoring* baterai dan modul SD *Card*. Pemrograman pada *board monitoring* baterai meliputi perancangan SOC, SOH, durasi, dan intensitas pengecasan. Sedangkan pemrograman pada SD *Card* meliputi penyimpanan data yang dikirimkan oleh *board monitoring* baterai.

3.3.1 Perancangan Perangkat Lunak SOC dan SOH

Pada proses estimasi kapasitas (SOC) dan kesehatan baterai (SOH), terdapat dua sensor arus ACS 758 yang dihubungkan ke *board monitoring* baterai. Masing-masing sensor tersebut berfungsi untuk mengukur arus pada saat pengecasan dan pengosongan arus baterai DPV. Pemrograman sensor arus ACS 758 dimulai dengan inisialisasi variabel, baudrate, dan *interrupt timer*. Pada pemrograman tersebut inisialisasi variabel diletakkan secara global agar dapat diakses secara

keseluruhan program. Pada pemrograman tersebut juga ditetapkan nilai baudrate sebesar 38000 bps, nilai tersebut digunakan untuk melakukan komunikasi serial antara modul SD *Card* dengan *board monitoring* baterai. Setelah itu program dilanjutkan dengan inisialisasi timer interrupt yang digunakan untuk menentukan nilai waktu penyamplingan pada saat pengambilan data.

Data yang diambil dari sensor masih berupa konversi dari nilai analog menuju digital atau Analog to Digital Converter (ADC), data ADC tersebut diolah menggunakan *digital low pass filter* untuk mengurangi *noise* seperti persamaan 3.1.

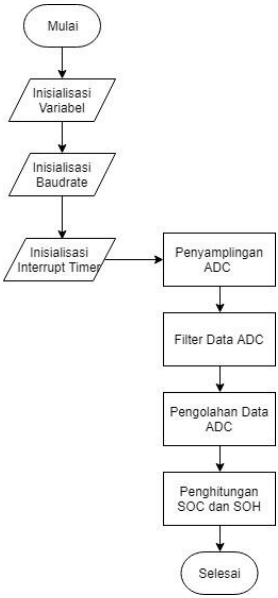
$$Filter_t = (0.9 \times Data) + (0.1 \times Filter_{t-1}) \quad (3.1)$$

$Filter_t$ merupakan data yang telah difilter, Data merupakan nilai ADC sebelumnya dan $Filter_{t-1}$ adalah nilai ADC baru. Untuk merubah nilai ADC menjadi arus, perlu dilakukan pengkonversian nilai ADC menjadi arus. Pengubahan nilai ADC menjadi arus menggunakan persamaan 3.2.

$$\text{Arus} = \frac{\left(\frac{\text{ADC}}{1023}\,5000\right) - 2500}{40} \quad (3.2)$$

Setelah itu untuk mendapatkan nilai arus, nilai ADC terlebih dahulu diubah menjadi nilai tegangan. Mikrokontroler ATmega 328P mempunyai resolusi pembacaan ADC 10 bit, sehingga ADC maksimum yang dapat dihasilkan ATmega 328P adalah 2^{10} = 1023. Nilai yang dihasilkan oleh ATmega 328P kemudian dibagi dengan nilai ADC maksimum dan dikalikan dengan 5000 untuk menghasilkan nilai tegangan dalam satuan milivolt. Setelah itu, untuk mengubah nilai tegangan menjadi arus. Nilai tegangan yang didapatkan dikurangi dengan nilai *offset* sensor sebesar 2500 dan dibagi 40 yang merupakan nilai sensivitas sensor.

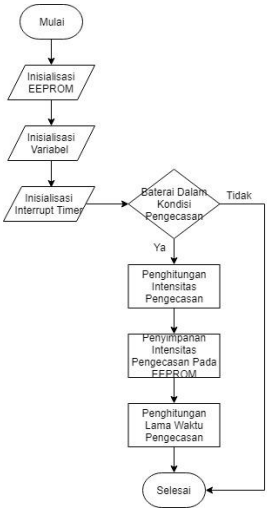
Setelah didapatkan nilai arus, nilai yang didapatkan kemudian dijumlahkan hingga proses *discharge* / *charge* berakhir. Data yang diterima oleh mikrokontroler merupakan sinyal diskrit, sehingga nilai arus yang didapatkan harus dikalikan dengan waktu *sampling* sebelum nilai dijumlahkan. *Flowchart* perancangan program SOC dan SOH dapat dilihat pada Gambar 3.8.



Gambar 3.8 *Flowchart* Perancangan Program SOC dan SOH

3.3.2 Perancangan Perangkat Lunak Durasi dan Intensitas Pengecasan Baterai

Pemrograman durasi dan intensitas pengecasan pada baterai dimulai dengan inisialisasi EEPROM dan dilanjutkan dengan inisialisasi variabel yang digunakan. Pemrograman dilanjutkan dengan inisialisasi *interrupt timer* yang digunakan untuk melakukan penghitungan lama waktu pengecasan. *Charger* baterai DPV akan terhubung dengan *board monitoring* baterai, ketika baterai tersebut dilakukan pengisian, *board monitoring* baterai akan mendeteksi melalui pin digital mikrokontroler. Setelah itu, program akan melakukan penghitungan durasi lama waktu pengecasan menggunakan fungsi *interrupt timer,* fungsi *interrupt timer* telah disediakan mikrokontroler. Untuk nilai intensitas pengecasan akan disimpan pada EEPROM agar dapat menyimpan nilai akhir pengecasan. *Flowchart* perancangan program durasi dan intensitas pengecasan dapat dilihat pada Gambar 3.9.



Gambar 3.9 *Flowchart* Perancangan Program Durasi dan Intensitas Pengecasan Baterai

3.3.3 Perancangan Perangkat Lunak Integrasi *Board* Pengukuran Baterai

Program pengukuran SOC dan SOH, serta durasi dan intensistas pengecasan pada baterai diintegrasikan menjadi sebuah *sketch* program Arduino agar dapat bekerja secara bersamaan. Pada program yang telah diintregasikan, setiap data yang diterima oleh *board* pengukuran baterai akan diolah terlebih dahulu oleh *board monitoring* baterai. Data yang telah diolah kemudian akan dikirimkan pada modul SD *Card* sebagai tempat penyimpanan data. Pada proses pengiriman data antara *board* baterai dengan modul SD *Card* tersebut menggunakan komunikasi serial untuk melakukan pengiriman data. Data yang disimpan pada modul SD *Card* akan disimpan dalam bentuk file excel.

3.4 Perancangan Perangkat Lunak Modul SD *Card*

Dalam melakukan perancangan perangkat lunak modul SD *Card*, pemrograman dimulai dengan inisialisasi library <SD.h> untuk melakukan pengaksesan *library* SD *Card* dan <SPI.h> untuk melakukan

komunikasi SPI antara mikrokontroler dengan SD *Card* slot, dan dilanjutkan dengan inisialisasi variabel – variabel yang digunakan untuk menyimpan nilai atau data. Pada pemrograman modul SD *Card* ini, nilai transfer data atau baudrate ditetapkan sebesar 38400 bps.

Program dilanjutkan dengan SD.open() yang digunakan untuk membuat atau membuka file dalam bentuk excel. Setelah itu, data yang diterima pada komunikasi serial akan terlebih dahulu diolah dan diseleksi menggunakan fungsi switch case. Data yang telah diseleksi kemudian akan disimpan pada file excel menggunakan fungsi myFile.println().

Halaman ini sengaja dikosongkan

BAB 4
PENGUJIAN DAN ANALISIS

4.1 Pengujian Sensor Arus ACS 758

Pengujian sensor arus dilakukan sesuai dengan Gambar 3.3. Pada pengujian tersebut, dilakukan pengambilan data arus dari sensor ACS 758 dan clamp meter. Pengujian dilakukan untuk mengetahui bahwa sensor bekerja sebagaimana mestinya. Hasil pengujian dapat dilihat pada Tabel 4.1.

Tabel 4. 1 Data Validasi Sensor Arus ACS 758

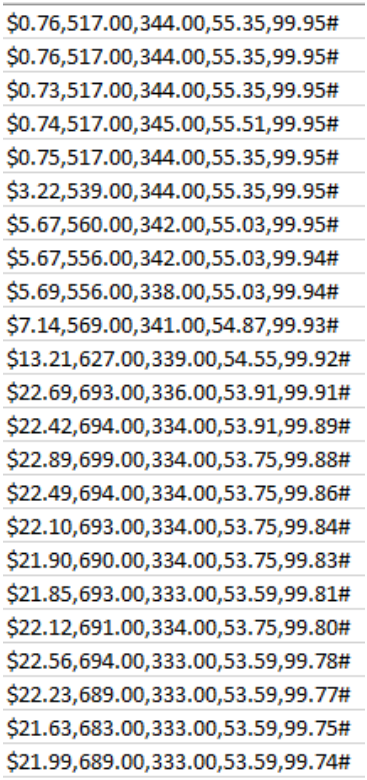
Arus Clamp Meter (A)	Arus Sensor ACS 758 (A)
0.5	0.51
1	1.03
1.5	1.48
2.5	2.48
3	2.98

Tabel 4.1 menunjukkan data perbandingan antara arus yang diukur menggunakan clamp meter dan sensor ACS 758. Dari hasil yang diperoleh, didapatkan nilai RMSE (*Root Mean Square Error*) sebesar 0.02273. Nilai RMSE didapatkan menggunakan persamaan 2.5, nilai arus yang terukur oleh clamp meter dijadikan sebagai masukan data sebenarnya sedangkan data dari sensor dijadikan sebagai data hasil estimasi. Dengan hasil ini maka dapat disimpulkan bahwa sensor menunjukkan hasil yang akurat.

4.2 Hasil Penyimpanan Data pada Modul SD *Card*

Data arus, tegangan, dan kapasitas baterai yang dihasilkan oleh *board monitoring* baterai disimpan pada modul SD *Card*. Pada modul SD *Card*, data disimpan dalam bentuk file excel. Penyimpanan pada modul SD *Card* menggunakan komunikasi serial, untuk mengantisipasi hilangnya data pada komunikasi tersebut maka digunakan *header*

berupa simbol $ dan *footer* dengan simbol # pada saat pengiriman data. Hasil penyimpanan data pada file excel dapat dilihat pada Gambar 4.1.

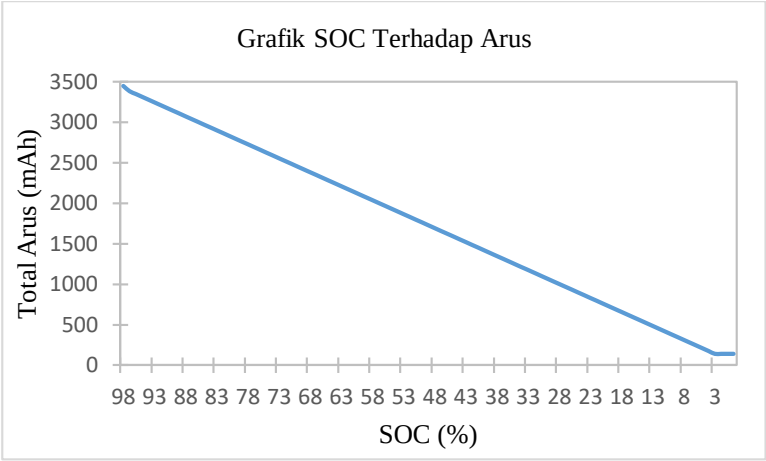


Gambar 4.1 Hasil Penyimpanan Data pada Modul SD *Card*

4.3 Hasil Pengujian SOC pada Baterai 1 Sel

Dalam tahap ini, dilakukan pengujian SOC sesuai dengan persamaan 2.1. Pengujian dilakukan terhadap baterai 1 sel berjenis lithium ion. Pengujian baterai 1 sel tersebut menggunakan sensor ACS 758. Pengujian ini bertujuan untuk membuktikan metode yang digunakan serta untuk mengetahui kesesuaian antara arus yang masuk dan keluar baterai sebelum melakukan pengujian pada baterai DPV.

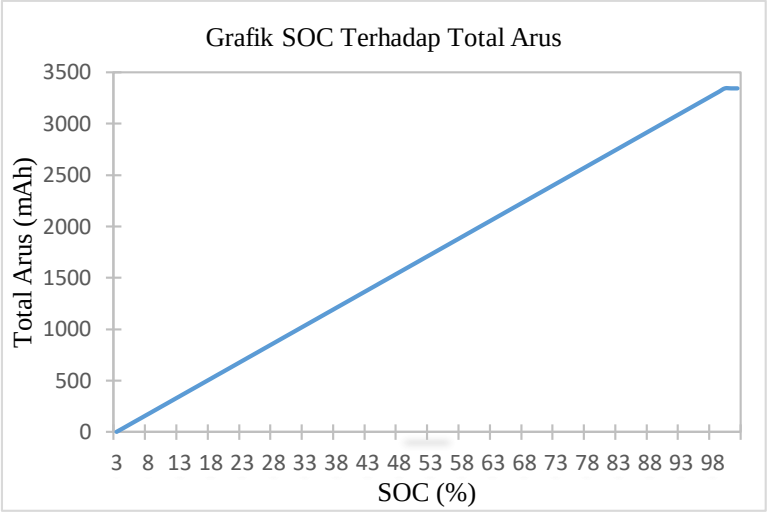
4.3.1 Pengosongan Arus Baterai (*Discharging*)



Gambar 4.2 Grafik SOC dan Arus *Discharging* Baterai 1 Sel

Pada pengujian ini didapatkan karakteristik baterai ketika kondisi *discharging*, nilai SOC pada Gambar 4.2 berada pada tingkat presentase 98%. Nilai SOC tersebut tidak dapat mencapai presentase 100% karena nilai tegangan baterai pada kondisi sebelum diberikan beban bernilai kurang dari 4.2 volt. Berkurangnya tegangan baterai sebelum diberikan beban dapat disebabkan karena penurunan kesehatan baterai. Penurunan kesehatan baterai dapat dikarenakan karena adanya penggunaan dari baterai yang mempengaruhi berkurangnya siklus / masa pakai baterai. Berkurangnya masa pakai baterai dapat menyebabkan baterai mengalami penurunan kapasitas, selain itu penurunan kapasitas dapat disebabkan karena kondisi pengecasan yang berlebihan (*overcharging*) atau pengosongan yang berlebihan (*overdischarging*). Pada pengujian ini, ketika baterai mendekati batas bawah tegangan. Arus yang mengalir pada beban semakin berkurang. Semakin berkurangnya arus menyebabkan penurunan kapasitas (SOC) semakin sedikit. Hal tersebut dapat dilihat pada Gambar 4.2 ketika SOC mencapai 3%. Pada kondisi 3% tersebut lampu yang terhubung dengan baterai berada dalam kondisi sangat redup. Pada pengujian ini, keseluruhan arus yang keluar bernilai sebesar 3.308 mAh. Nilai arus keluar akan dibandingkan ketika kondisi melakukan pengisian (*charging*).

4.3.2 Pengisian Arus Baterai (*Charging*)



Gambar 4.3 Grafik SOC dan Arus *Charging* Baterai 1 Sel

Gambar 4.3 merupakan grafik ketika kondisi pengisian pada baterai 1 sel. Kondisi awal pengujian ini diberikan nilai 3%, dimana nilai tersebut merupakan hasil terakhir ketika kondisi pengosongan arus (*discharging*). Pada pengujian ini didapatkan total arus sebebsar 3.346 mAh, setelah baterai berada pada kondisi penuh nilai arus akan tetap pada kondisi kostan. Hal tersebut dapat dilihat pada grafik setelah mencapai titik 98%. Selisih antara nilai arus masuk dan keluar adalah 38 mAh, dimana arus ketika kondisi *charging* lebih banyak dari pada arus yang terekam ketika kondisi *discharging*. Pada pengujian ini dilakukan menggunakan metode Coloumb Counting, yaitu penjumlahan arus terhadap waktu. Selisih tersebut dapat dikarenakan penggunaan metode yang mengharuskan penjumlahan pada setiap data yang dihasilkan sensor. Setiap data yang dihasilkan sensor mempunyai nilai eror yang berbeda-beda setiap waktunya. Eror yang terdapat pada sensor tersebut diakumulasikan sehingga didapatkan eror yang cukup banyak selama berjalannya waktu atau periode *sampling* berlangsung.

4.4 Pengujian SOC pada Baterai DPV

Berdasarkan hasil pengujian pada baterai 1 sel, sistem yang digunakan telah berjalan dengan baik sehingga pengujian dilanjutkan menggunakan baterai DPV.

4.4.1 Data SOC ketika Pengosongan Arus (*Discharging)*

Dalam pengujian ini, dilakukan pengosongan arus pada baterai dengan beban motor DPV menggunakan persamaan 1. Sebelum melakukan pengosongan arus, baterai terlebih dahulu dilakukan pengecasan hingga mencapai kondisi penuh. Setelah itu, dilakukan pengosongan arus secara konstan pada baterai menggunakan beban motor dengan nilai arus ± 20 A.

Tabel 4. 2 *Sampling* Data SOC pada Kondisi Pengosongan Arus

No.	SOC (%)	Sisa Kapasitas (mAh)
1	98	39200
2	95	38169
3	90	36223
4	85	34275
5	80	32331
6	75	30378
7	70	28429
8	65	26478
9	60	24529
10	55	22579
11	50	20630
12	45	18728
13	40	16828
14	35	14932
15	30	13036
16	25	11142
17	20	9245

No.	SOC (%)	Sisa Kapasitas (mAh)
18	15	7343
19	10	5447
20	5	3547
21	2	1643

Dari data hasil pengujian yang terdapat pada Tabel 4.2, nilai arus yang terukur oleh sensor adalah 38357 mAh. Sedangkan nilai yang didapatkan menggunakan penghitungan manual berdasarkan avometer sebesar 39616 mAh. Dari kedua data yang didapatkan terdapat selisih atau eror sebesar 3.1%. Berdasarkan data pada Tabel 4.2, hasil akhir estimasi kapasitas baterai, terdapat sisa arus sebesar 1643 mAh ketika kondisi 2%. Nilai kesalahan tersebut disebabkan karena kesalahan pembacaan oleh sensor arus.

4.4.2 Data SOC ketika Pengisian Arus (*Charging*)

Tabel 4. 3 *Sampling* Data SOC pada Kondisi Pengisian Arus

No.	SOC (%)	Kapasitas (mAh)
1	2	1643
2	5	2249
3	10	4299
4	15	6349
5	20	8399
6	25	10448
7	30	12499
8	35	14551
9	40	16599
10	45	18698
11	50	20799
12	55	22898
13	60	24997
14	65	27097
15	70	29197
16	75	31298

No.	SOC (%)	Kapasitas(mAh)
17	80	33397
18	85	35502
19	90	37599
20	95	39701
21	101	41798

Tabel 4.3 merupakan *sampling* data SOC pada saat pengisian arus baterai, pada saat pengisian baterai jumlah arus yang dibaca sensor adalah 41798 mAh. Sedangkan nilai yang didapatkan menggunakan penghitungan manual berdasarkan avometer sebesar 39988 mAh. Terdapat selisih nilai / eror sebesar 4.5%. Dari data Tabel 4.3, nilai maksimum dari pembacaan sensor melebihi kapasitas baterai. Nilai kesalahan tersebut disebabkan karena terjadi kesalahan pembacaan / eror ketika pengambilan data pada sensor arus, pengambilan data sensor arus dilakukan setiap detik. Banyaknya pengambilan data selama baterai dalam kondisi *charging* menyebabkan terjadinya akumulasi kesalahan pembacaan. Akumulasi kesalahan pembacaan tersebut menimbulkan ketidakakuratan ketika estimasi nilai akhir.

4.5 Nilai Kesehatan Baterai

Berdasarkan data yang diperoleh pada saat *discharge*, Nilai SOH didapatkan dengan membandingkan nilai keseluruhan kapasitas arus baterai yang terukur dengan nilai nominal baterai. Baterai lithium iron phosphate yang digunakan DPV mempunyai nilai kapasitas nominal sebesar 40 Ah. Pada pengukuran keseluruhan kapasitas baterai, didapatkan nilai total arus pada saat pengosongan arus atau *discharging* baterai sebesar 37557 mAh, nilai tersebut didapatkan dengan mengurangi nilai kapasitas awal dengan kapasitas akhir baterai yang terdapat pada Tabel 4.2.

Berdasarkan nilai yang didapatkan pada saat pengukuran baterai, maka nilai SOH dapat dihitung menggunakan persamaan 2 sehingga menghasilkan nilai :

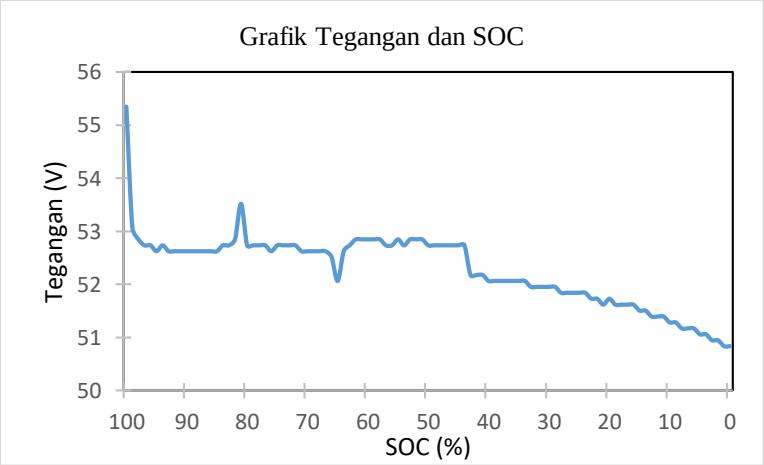
$$\mathrm{SOH} = \frac{37557}{40000} \times 100\%$$

Dari perbandingan kapasitas nilai yang terukur dengan nilai nominal baterai, didapatkan nilai kesehatan baterai sebesar 93.89%. Berdasarkan nilai tersebut, dapat disimpulkan bahwa baterai masih dalam kondisi baik.

4.6 Relasi Tegangan dan SOC

4.6.1 Tegangan dan SOC ketika *Discharge*

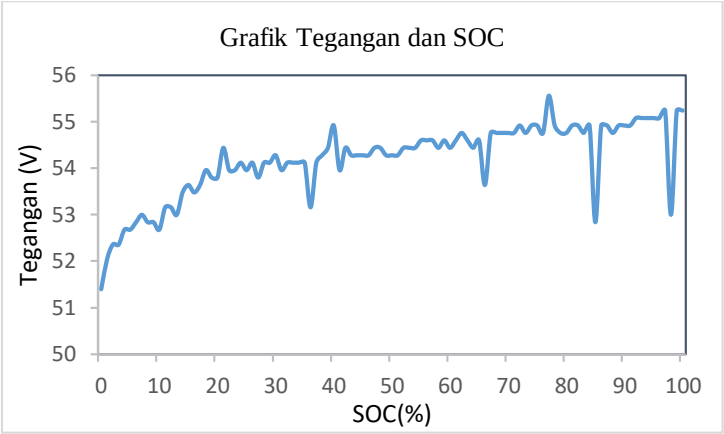
Pengambilan data tegangan dan SOC digunakan untuk mengetahui relasi antara kedua paramater pada saat kondisi baterai diberikan beban. Pada saat kondisi arus diberikan beban dengan sebesar 20A, baterai tersebut mengalami penurunan tegangan (*drop voltage*) hingga 2.44V. selain itu pengambilan data tersebut digunakan untuk mengetahui nilai tegangan minimum pada saat BMS melakukan *cut off*. Dengan beban arus yang dikeluarkan baterai sebesar 20 A, nilai tegangan baterai pada saat BMS DPV akan melakukan *cut off* adalah 50.82 V.



Gambar 4.4 Grafik SOC dengan Tegangan pada Kondisi Pengosongan Arus Baterai

4.6.2 Tegangan dan SOC ketika kondisi Pengisian Baterai

Pengambilan data tegangan dengan SOC digunakan untuk melihat relasi antara tegangan dan SOC ketika baterai berada pada kondisi tanpa beban dan digunakan untuk melihat tegangan maksimum pada saat SOC mencapai kondisi 100%. Pada saat melakukan pengisian arus baterai dan kondisi baterai mencapai 100% didapatkan nilai tegangan sebesar 56.3V namun arus tersebut turun ketika *charger* dilepas dan mengalami penurunan hingga nilai tegangan berada di nilai 55.44V.



Gambar 4.5 Grafik SOC dengan Tegangan pada Kondisi Pengisian Arus Baterai

Halaman ini sengaja dikosongkan

BAB 5
PENUTUP

5.1 Kesimpulan

Berdasarkan pengujian dan analisis yang telah dilakukan, estimasi kapasitas dan indikator kesehatan pada baterai menggunakan metode coulomb counting mempunyai kekurangan terdapat akumulasi kesalahan oleh pembacaan sensor. Pada saat kondisi discharging terdapat nilai kesalahan / eror sebesar 3.1% dan ketika *charging* sebesar 4.5% dari penghitungan kapasitas secara manual. Selain itu, baterai yang diuji masih mempunyai kondisi yang bagus dengan nilai kesehatan sebesar 93.89%.

5.2 Saran

Untuk pengembangan selanjutnya, dapat ditambahkan mengenai tampilan dari kapasitas baterai menggunakan perhitungan SOC pada LCD DPV dan penambahan rangkaian pengukur tegangan tiap sel baterai. Hal tersebut bertujuan agar pengguna dapat mengetahui tingkat kapasitas baterai dan ditambahkannya pengukur tegangan tiap sel baterai yang bertujuan untuk memonitor nilai tegangan sel baterai. Selain itu, dalam pembuatan PCB ukuran *board* juga perlu dibuat sekecil mungkin agar *board* dapat dimasukkan pada BMS DPV.

Halaman ini sengaja dikosongkan

DAFTAR PUSTAKA

[1] H. Anam *et al.*, "Recent Advances in Diver Propulsion Vehicle (DPV) Ganendra RI-1 : Design , Analysis , Manufacture and Performance," no. February 2018, 2016.

[2] K. W. E. Cheng, B. P. Divakar, H. Wu, K. Ding, and H. F. Ho, "Battery-management system (BMS) and SOC development for electrical vehicles," *IEEE Trans. Veh. Technol.*, vol. 60, no. 1, pp. 76–88, 2011.

[3] B. A. Enache, M. E. Alexandru, and L. M. Constantinescu, "A LiFePO<inf>4</inf> battery *discharge* simulator for EV applications - Part 2: Developing the battery simulator," *2015 9th Int. Symp. Adv. Top. Electr. Eng. ATEE 2015*, no. May, pp. 883–888, 2015.

[4] X. Wang, P. Adelmann, and T. Reindl, "Use of LiFePO4batteries in stand-alone solar system," *Energy Procedia*, vol. 25, pp. 135–140, 2012.

[5] V. M. Dileepan and J. Jayakumar, "Performance analysis of lithium batteries," *Proc. IEEE Int. Conf. Innov. Electr. Electron. Instrum. Media Technol. ICIEEIMT 2017*, vol. 2017-Janua, no. 978, pp. 330–333, 2017.

[6] P. A. Topan, M. N. Ramadan, G. Fathoni, A. I. Cahyadi, and O. Wahyunggoro, "*State Of Charge* (SOC) and *State Of Health* (SOH) estimation on lithium polymer battery via Kalman filter," *Proc. - 2016 2nd Int. Conf. Sci. Technol. ICST 2016*, no. 2, pp. 93–96, 2017.

[7] L. Buccolini, A. Ricci, C. Scavongelli, G. Demaso-Gentile, S. Orcioni, and M. Conti, "Battery Management System (BMS) simulation environment for electric vehicles," *EEEIC 2016 - Int. Conf. Environ. Electr. Eng.*, pp. 2–7, 2016.

[8] P. . Topan, "Estimasi Parameter Model Baterai, *State Of Charge*, dan *State Of Health* pada Baterai Lithium Polymer," *Director*, vol. 15, p. 74, 2018.

[9] G. Recktenwald, "Voltage Dividers and Potentiometers," pp. 1–8, 2012.

[10] A. Microsystems, "Not for New Design ACS758xCB," 2017.

[11] D. Lab, "Micro SD Memory *Card* interface Table of Contents."

[12] C. Cossar and I. Young, "Ananaba, Kemjika. Design and Implementation of a Buck Converter," pp. 1–72.

[13] D. P. Frame and S. Box, “Datasheet 2A 150KHz 40V Buck DC to DC Converter Features XL1509 General Description Datasheet 2A 150KHz 40V Buck DC to DC Converter Pin Configurations,” pp. 1–13.
[14] A. Hussain, M. Hammad, K. Hafeez, and T. Zainab, “Programming a Microcontroller,” *Int. J. Comput. Appl.*, vol. 155, no. 5, pp. 21–26, 2016.
[15] G. Gridling, “Other Microcontrollers,” *Introd. to Microcontrollers*, pp. 379–414, 2007.
[16] Atmel, “ATmega328 / P,” *Datasheet*, p. 442, 2016.
[17] M. Fezari and A. Al Dahoud, “Integrated Development Environment ‘ IDE ’ For Arduino,” no. October, 2018.
[18] R. D. D. Basics, “Programming With Arduino,” pp. 23–60.
[19] M. Aasvik, “Potentiometers with Arduino and Filtering,” *Nor. Creat.*

LAMPIRAN

Lampiran 1 : *Sketch* program *Board Monitoring* Baterai

```
//BoardConfig
//Pin ADC 0 = multiplex A
//Pin ADC 1 = multiplex B
//Pin ADC 2 = multiplex C
//Pin ADC 3 = kosong
//Pin ADC 4 = SDA
//Pin ADC 5 = SCL
//Pin ADC 6 = Sensor ACS Discharge
//Pin ADC 7 = Sensor ACS Charge

#include <EEPROM.h>
#define pin 13
#define add 0
bool flagtime = false;
byte flag,
     getdata,
     hitung = 0,
     sekarang = 0,
     terakhir = 0,
unsigned int ms;
float adcAwal1, adcAwal2,
      adcFilter1, adcFilter2,
      adcBaru1, adcBaru2,
      adcFix1, adcFix2,
double timer,
       waktunyala,
       arusTotal1, arusTotal2,
       arusHitung1, arusHitung2,
       adcvoltage, voltage;

void setup() {

  Serial.begin(38400);

  TCCR1A = (0 << COM1A1) | (0 << COM1A0) | (0 <<
COM1B1) | (0 << COM1B0) | (1 << WGM11) | (0 <<
WGM10);
  TCCR1B = (0 << ICNC1) | (0 << ICES1) | (1 << WGM13)
| (1 << WGM12) | (0 << CS12) | (1 << CS11) | (1 <<
CS10);
  TCNT1H = 0x00;
  TCNT1L = 0x00;
  ICR1H = 0x00;
  ICR1L = 0xF9;
  OCR1AH = 0x00;
  OCR1AL = 0x00;
```

```
  OCR1BH = 0x00;
  OCR1BL = 0x00;

  // Timer/Counter 1 Interrupt(s) initialization
  TIMSK1 = (1 << ICIE1) | (0 << OCIE1B) | (0 <<
  OCIE1A) | (0 << TOIE1);

  adcAwal1   = analogRead(A0);
  adcFilter1 = adcAwal1;

  adcAwal2   = analogRead(A0);
  adcFilter2 = adcAwal2;
}


void loop() {
  penghitungan();
  if (flag == 1)
  {
    sensorArus Charge(); //lama //sipp
    sensorArus Discharge(); //baru //kabel gesrek
    flag = 0;
  }
}

void sensorArus Discharge() //Sensor Lama
{
  double arusHitung1 = (((((adcFix1 / 1023.0) * 5) -
  2.5) / 0.04)) - 0.600645; 566; // 0.455645;
  //0.280645;

  arusTotal1 = arusTotal1 + arusHitung1;
  double SOC1 = 0.0 + (arusTotal1 / 144000.0) * 100;
  //40A x 3600s = 12060 x 4 = 48240

  String kirimArus1 = String(arusHitung1, 1);
  voltage = (adcvoltage / 1023.0) * 5 *  (99800 +
  3150) / 3150 - 1.95; //(109610)/9810;

  //  $xx.xx, xxx.xx, xxx.xx, xx.xx, xxx.xx#
  //   123456 7891234 5678912 345678 912345

  Serial.print("$");
  Serial.print(arusHitung1);
  Serial.print(",");
  Serial.print(adcBaru1);
  Serial.print(",");
  Serial.print(adcvoltage);
  Serial.print(",");
```

```
Serial.print(voltage);
  Serial.print(",");
  Serial.print(SOC1);
  Serial.println("# ");
}

void sensorArusCharge() //SensorBaru
{
  double arusHitung2 = (((((adcFix2 / 1023.0) * 5) -
  2.5) / 0.04)) + 0.955645;
  arusTotal2 = arusTotal2 + arusHitung2;
  double SOC2 = 0.0 + (arusTotal2 / 48240.0) * 100;
  String kirimArus2 = String(arusHitung2, 1);
  Serial.print("$");
  Serial.print(arusHitung2);
  Serial.print(",");
  Serial.print(adcBaru2);
  Serial.print(",");
  Serial.println(SOC2);
  Serial.println("#");
}

void penghitungan()
{
  sekarang = digitalRead(pin);
  if (sekarang != terakhir)
  {
    if ( sekarang == HIGH)
    { getdata = EEPROM.read(add);
      hitung = getdata;
      hitung++;
      Serial.print ("\n\t\t\t\t$ON Chg ");
      Serial.print(hitung);
      Serial.println ("x#");
      EEPROM.update(add, hitung);
      flagtime = true;
    }
    else
    {
      Serial.print("\n\t\t\t\t$OFF Tim ");
      flagtime = false;
      waktunyala = timer / 1000;
      Serial.print(waktunyala);
      Serial.println("s#");
    }  }
  terakhir = sekarang;}

ISR(TIMER1_CAPT_vect)
{
```

```
if (flagtime == true)
  {
    timer++;
  }

  // Sampling Sensor Arus Charge
  adcBaru1 = analogRead(A7);
  adcFilter1 = (0.9 * adcFilter1) + (0.1 * adcBaru1);
  adcvoltage =  analogRead(A6);

  // Sampling Sensor Arus Discharge
  adcBaru2 = analogRead(A0);
  adcFilter2 = (0.9 * adcFilter2) + (0.1 * adcBaru2);

  ms++;
  if (ms == 1000)
  {
    flag = 1;
    adcFix1 = adcFilter1;
    adcFix2 = adcFilter2;
    ms = 0;
  }
}
```

Lampiran 2 : *Sketch* program SD *Card*

```
#include <SD.h>
#include <SPI.h>
File myFile;

String dataChar[17];
String tempString[17];
String data[17];

byte menucase;
bool tulis[17];
byte i;

void setup() {

Serial.begin(38400);
pinMode(pinCS, OUTPUT);
tulis[17]= false;
if (SD.begin())
  {
   Serial.println("SD card Siap.\n");
}
```

```
else
    {
      Serial.println("SD card initialization
failed");
      return;
    }

  //  --------- Create/Open file ---------
  myFile = SD.open("test.csv", FILE_WRITE);


  //   --------- if the file opened okay, write to
  if (myFile) {

    ////////    --------- Write to file ---------

    myFile.println("hehehe4444");

    myFile.close(); // close the file

  }
  -------- - if the file didn't open, print an error:
  else {
  Serial.println("error opening test.csv \n");}

// --------- Reading the file ---------
myFile = SD.open("test.csv");
  if (myFile) {
  Serial.println("Isine File Excel :");
  // Reading the whole file
  while (myFile.available()) {
  Serial.write(myFile.read());
}
  myFile.close();
}
else {
  Serial.println("error opening test.csv");
}
SD.remove("test.csv"); // remove is a function in SD
library to delete a file
  if (SD.exists("test.csv"))
  { //If the file still exist display message exist
  Serial.println("File csv ada."); } }
else
  { //If the file was successfully deleted display
message that the file was already deleted.
  Serial.println("\n --- The test.csv doesn't
exist.");
} }
```

```
else
  { //If the file was successfully deleted display
message that the file was already deleted.
  Serial.println("\n --- The test.csv doesn't exist.");
} }
void loop() {}

  void serialEvent()
  {
  while(Serial.available())
  { char serial = Serial.read();
  switch(menucase){
  case 0:
  if(serial == '$')
  {menucase = 1;}
  break;

  case 1:
  if(serial == '$' )
  {menucase = 1;}
else
  {dataChar[0]=serial; menucase=2;} //1
  break;

  case 2:
  if(serial == '$' )
  {menucase = 1;}
  else if(serial == '#')
  {
  for(i=0;i<1;i++){tempString[0]+=dataChar[i];}
  data[0]= tempString[0]; tempString[0]="";
  menucase = 0; excel(); tulis[0]=true
}
else
  {dataChar[1]=serial; menucase=3;} //2
  break;

  case 3:
  if(serial == '$' )
  {menucase = 1;}
  else if(serial == '#')
  {
  for(i=0;i<2;i++){tempString[1]+=dataChar[i];}
  data[1]= tempString[1]; tempString[1]="";
  menucase = 0; excel(); tulis[1]=true
}
else
  {dataChar[2]=serial; menucase=4;} //3
  break;
```

```
case 4:
  if(serial == '$' )
  {menucase = 1;}
  else if(serial == '#')
  {
  for(i=0;i<3;i++){tempString[2]+=dataChar[i];}
  data[2]= tempString[2]; tempString[2]="";
  menucase = 0; excel(); tulis[2]=true
}
  else
  {dataChar[3]=serial; menucase=5;} //4
  break;

  case 5:
  if(serial == '$')
  {menucase = 1;}
  else if(serial == '#')
  {
  for(i=0;i<4;i++){tempString[3]+=dataChar[i];}
  data[3]= tempString[3]; tempString[3]="";
  menucase = 0; excel(); tulis[3]=true }
  else
  {dataChar[4]=serial; menucase=6;} //5
  break;

  case 6:
  if(serial == '$' )
  {menucase = 1;}
  else if(serial == ',')
  {
  {dataChar[5]=serial; menucase=7;} }
  else if(serial == '#')
  {
  for(i=0;i<5;i++){tempString[4]+=dataChar[i];}
  data[4]= tempString[4]; tempString[4]="";
  menucase = 0; excel(); tulis[4]=true }

  else
  {dataChar[5]=serial; menucase=7;} //6
  break;
  case 7:
  if(serial == '$' )
  {menucase = 1;}
  else if(serial == '#')
  {
  for(i=0;i<6;i++){tempString[5] += dataChar[i];}
  data[5]= tempString[5]; tempString[5]="";
  menucase = 0; excel(); tulis[5]=true;
}
```

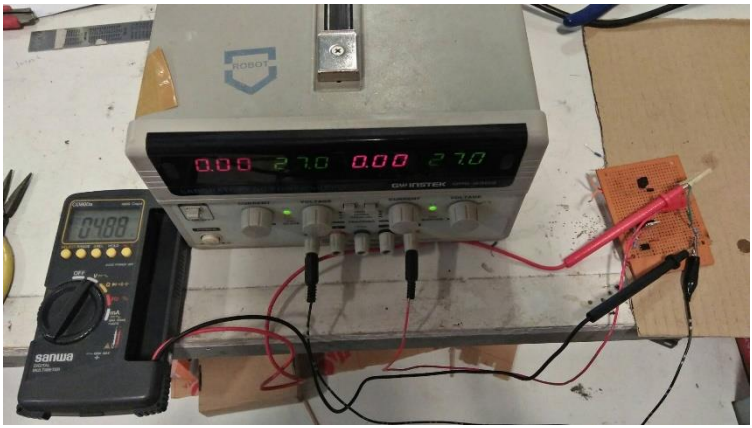
```
else
  {dataChar[6]=serial; menucase=8;} //7
  break;

  case 8:
  if(serial == '$' )
  {menucase = 1;}
  else if(serial == '#')
  {
  for(i=0;i<7;i++){tempString[6] += dataChar[i];}
  data[6]= tempString[6]; tempString[6]="";
  menucase = 0; excel(); tulis[6]=true;
}
  else
  {dataChar[7]=serial; menucase=9;} //8
  break;

  case 9:
  if(serial == '$' )
  {menucase = 1;}
  else if(serial == '#')
  {
  for(i=0;i<8;i++){tempString[7] += dataChar[i];}
  data[7]= tempString[7]; tempString[7]="";
  menucase = 0; excel(); tulis[7]=true;
}
  else
  {dataChar[8]=serial; menucase=10;} //9
  break;

  case 10:
  if(serial == '$' )
  {menucase = 1;}
  else if(serial == '#')
  {
  for(i=0;i<9;i++){tempString[8] += dataChar[i];}
  data[8]= tempString[8]; tempString[8]="";
  menucase = 0; excel(); tulis[8]=true;
}
  else if(serial == ',')
  {dataChar[9]=serial; menucase=11;
}
  else
  {dataChar[9]=serial; menucase=11;} //10
  break;

  case 11:
  if(serial == '$' )
  {menucase = 1;}
  else if(serial == '#')
```

```
{
  for(i=0;i<10;i++){tempString[9] += dataChar[i];}
  data[9]= tempString[9]; tempString[9]="";
  menucase = 0; excel(); tulis[9]=true;
}
  else if(serial == ',')
  {
  dataChar[10]=serial; menucase=12;}
  else
  {dataChar[10]=serial; menucase=12;} //11
  break;

  case 12:
  if(serial == '$' )
  {menucase = 1;}
  else if(serial == '#')
  {
  for(i=0;i<11;i++){tempString[10] += dataChar[i];}
  data[10]= tempString[10]; tempString[10]="";
  menucase = 0; excel(); tulis[10]=true;
}
  else if(serial == ',')
  {dataChar[11]=serial; menucase=13;}
  else
  {dataChar[11]=serial; menucase=13;} //12
  break;

  // after comma

  case 13:
  if(serial == '$' )
  {menucase = 1;}
  else if(serial == '#')
  {
  for(i=0;i<12;i++){tempString[11] += dataChar[i];}
  data[11]= tempString[11]; tempString[11]="";
  menucase = 0; excel(); tulis[11]=true;
}
  else
  {dataChar[12]=serial; menucase=14;} //13
  break;

  case 14:
  if(serial == '$' )
  {menucase = 1;}
  else if(serial == '#')
  {
  for(i=0;i<13;i++){tempString[12] += dataChar[i];}
  data[12]= tempString[12]; tempString[12]="";
```

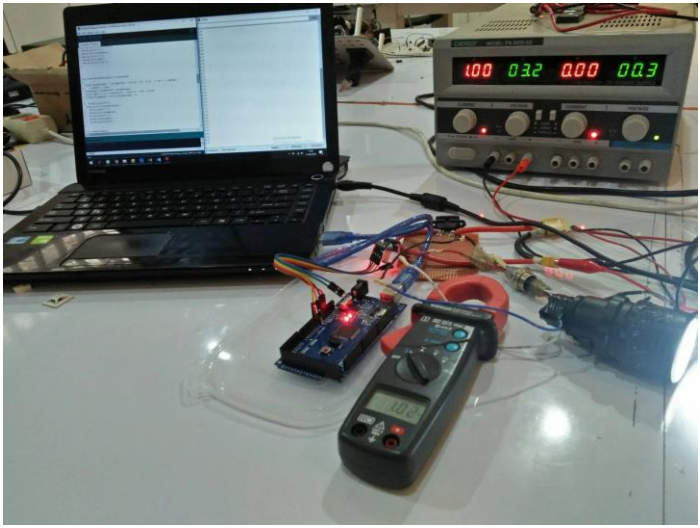
```
menucase = 0; excel(); tulis[12]=true;
}
  else
  {dataChar[13]=serial; menucase=15;} //14
  break;

  case 15:
  if(serial == '$' )
  {menucase = 1;}
  else if(serial == '#')
  {
  for(i=0;i<14;i++){tempString[13] += dataChar[i];}
  data[13]= tempString[13]; tempString[13]="";
  menucase = 0; excel(); tulis[13]=true;
}
  else
  {dataChar[14]=serial; menucase=16;} //15
  break;

  case 16:
  if(serial == '$' )
  {menucase = 1;}
  else if(serial == '#')
  {
  for(i=0;i<15;i++){tempString[14] += dataChar[i];}
  data[14]= tempString[14]; tempString[14]="";
  menucase = 0; excel(); tulis[14]=true;
}
  else
  {dataChar[15]=serial; menucase=17;} //16
  break;

  case 17:
  if(serial == '$' )
  {menucase = 1;}
  else if(serial == '#')
  {
  for(i=0;i<16;i++){tempString[15] += dataChar[i];}
  data[15]= tempString[15]; tempString[15]="";
  menucase = 0; excel(); tulis[15]=true;
}
  else
  {dataChar[16]=serial; menucase=18;} //17
  break;

  case 18:
```

```
if(serial == '#')
  {
  for(i=0;i<17;i++){tempString[16] += dataChar[i];}
  data[16]= tempString[16]; tempString[16]="";
  menucase = 0;  excel(); tulis[16]=true;
}
  break;}}}

void excel()
  {
  myFile = SD.open("test.csv", FILE_WRITE);

  if (myFile) {

  if(tulis[0]==true)
  { myFile.println(data[0]); tulis[0] = false; }
  if(tulis[1]==true)
  { myFile.println(data[1]); tulis[1] = false; }
  if(tulis[2]==true)
  { myFile.println(data[2]); tulis[2] = false; }
  if(tulis[3]==true)
  { myFile.println(data[3]); tulis[3] = false; }
  if(tulis[4]==true)
  { myFile.println(data[4]); tulis[4] = false; }
  if(tulis[5]==true)
  { myFile.println(data[5]); tulis[5] = false; }
  if(tulis[6]==true)
  { myFile.println(data[6]); tulis[6] = false; }
  if(tulis[7]==true)
  { myFile.println(data[7]); tulis[7] = false; }
  if(tulis[8]==true)
  { myFile.println(data[8]); tulis[8] = false; }
  if(tulis[9]==true)
  { myFile.println(data[9]); tulis[9] = false; }
  if(tulis[10]==true)
  { myFile.println(data[10]); tulis[10] = false; }
  if(tulis[11]==true)
  { myFile.println(data[11]); tulis[11] = false; }
  if(tulis[12]==true)
  { myFile.println(data[12]); tulis[12] = false; }
  if(tulis[13]==true)
  { myFile.println(data[13]); tulis[13] = false; }
  if(tulis[14]==true)
  { myFile.println(data[14]); tulis[14] = false; }
  if(tulis[15]==true)
  { myFile.println(data[15]); tulis[15] = false; }
  if(tulis[16]==true)
  { myFile.println(data[16]); tulis[16] = false; }
  myFile.close(); } }
```

Lampiran 3 : Pengujian Rangkaian Pembagi Tegangan



Lampiran 4 : Validasi Sensor Arus

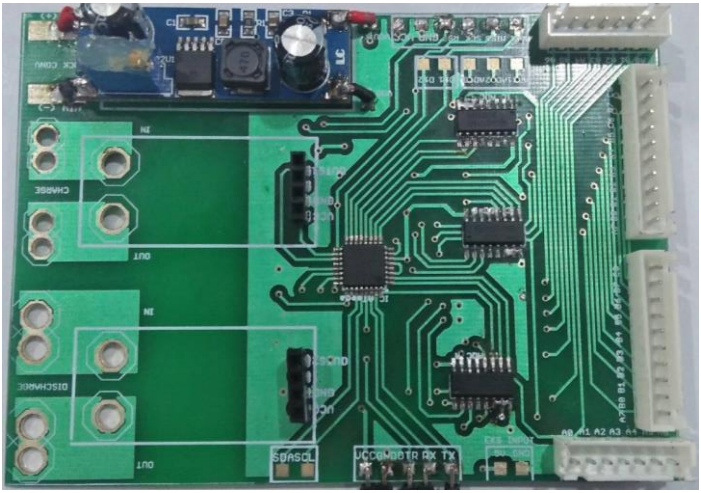


Lampiran 5 : Pengujian *Charging* Baterai DPV

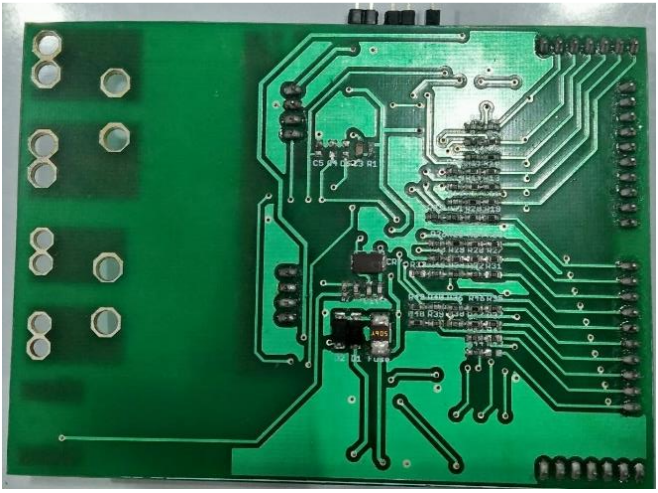
Lampiran 6 : Pengujian *Discharging* Pada Baterai DPV

Lampiran 7 : Beban Motor DPV

Lampiran 8 : Bagian Atas *Board Monitoring* Baterai



Lampiran 9 : Bagian Bawah *Board Monitoring* Baterai



Lampiran 10 : Data Penghitungan Manual kondisi *Discharging*

Arus (A)	Waktu (menit)	Ampere Second(As)
21,8	0	1890
21	1,5	2121
20,8	3,2	1722
20,5	92,1	105780
20	109	21600

Lampiran 11 : Data Penghitungan Manual kondisi *Charging*

Arus (A)	Waktu (menit)	Ampere Second(As)
10,4	2311	141876

Lampiran 12 : Data Penghitungan Manual kondisi *Discharging*

SOC (%)	Kapasitas (mAh)
98	3446
97	3382
96	3347
95	3313
94	3278
93	3243
92	3209
91	3174
90	3140
89	3106
88	3071
87	3036
86	3002
85	2967
84	2933
83	2899
82	2864
81	2829
80	2795
79	2760
78	2726
77	2691
76	2657
75	2622
74	2588
73	2553

SOC (%)	Kapasitas (mAh)
72	2519
71	2485
70	2450
69	2415
68	2381
67	2346
66	2312
65	2277
64	2243
63	2208
62	2174
61	2139
60	2105
59	2070
58	2036
57	2002
56	1967
55	1932
54	1898
53	1863
52	1829
51	1794
50	1760
49	1725
48	1691
47	1656
46	1622
45	1588

SOC (%)	Kapasitas (mAh)
44	1553
43	1518
42	1484
41	1450
40	1415
39	1380
38	1346
37	1311
36	1277
35	1242
34	1207
33	1174
32	1139
31	1104
30	1070
29	1035
28	1001
27	966
26	932
25	897
24	863
23	828
22	794
21	759
20	725
19	690
18	656
17	621

SOC (%)	Kapasitas (mAh)
16	587
15	552
14	518
13	483
12	449
11	414
10	380
9	345
8	311
7	276
6	242
5	207
4	173
3	138

Lampiran 13 : Data SOC dan Kapasitas Arus ketika *Charging* 1 Sel Baterai

SOC (%)	Kapasitas (mAh)
3	0
4	34
5	69
6	103
7	138
8	172
9	207
10	241
11	276

SOC (%)	Kapasitas (mAh)
12	310
13	345
14	379
15	414
16	448
17	483
18	517
19	552
20	586
21	621
22	655
23	690
24	724
25	759
26	793
27	828
28	862
29	897
30	931
31	966
32	1000
33	1035
34	1069
35	1104
36	1138
37	1173
38	1207
39	1242

SOC (%)	Kapasitas (mAh)
40	1276
41	1311
42	1345
43	1380
44	1414
45	1449
46	1483
47	1518
48	1552
49	1587
50	1621
51	1656
52	1690
53	1725
54	1759
55	1794
56	1828
57	1863
58	1897
59	1932
60	1966
61	2001
62	2035
63	2070
64	2104
65	2139
66	2173
67	2208

SOC (%)	Kapasitas (mAh)
68	2242
69	2277
70	2311
71	2346
72	2380
73	2415
74	2449
75	2484
76	2518
77	2553
78	2587
79	2622
80	2656
81	2691
82	2725
83	2760
84	2794
85	2829
86	2863
87	2898
88	2932
89	2967
90	3001
91	3036
92	3071
93	3105
94	3139
95	3174

SOC (%)	Kapasitas (mAh)
96	3208
97	3243
98	3277
99	3312
100	3346
100	3346

Lampiran 14 : Data Tegangan dan SOC ketika *Discharging* pada Baterai DPV

SOC (%)	Tegangan (V)
98	55,35
99	53,07
98	52,85
97	52,74
96	52,74
95	52,63
94	52,74
93	52,63
92	52,63
91	52,63
90	52,63
89	52,63
88	52,63
87	52,63
86	52,63
85	52,63
84	52,74

SOC (%)	Tegangan (V)
83	52,74
82	52,85
81	53,52
80	52,74
79	52,74
78	52,74
77	52,74
76	52,63
75	52,74
74	52,74
73	52,74
72	52,74
71	52,63
70	52,63
69	52,63
68	52,63
67	52,63
66	52,51
65	52,07
64	52,63
63	52,74
62	52,85
61	52,85
60	52,85
59	52,85
58	52,85
57	52,74
56	52,74

SOC (%)	Tegangan (V)
55	52,85
54	52,74
53	52,85
52	52,85
51	52,85
50	52,74
49	52,74
48	52,74
47	52,74
46	52,74
45	52,74
44	52,74
43	52,18
42	52,18
41	52,18
40	52,07
39	52,07
38	52,07
37	52,07
36	52,07
35	52,07
34	52,07
33	51,96
32	51,96
31	51,96
30	51,96
29	51,96
28	51,84

SOC (%)	Tegangan (V)
27	51,84
26	51,84
25	51,84
24	51,84
23	51,73
22	51,73
21	51,62
20	51,73
19	51,62
18	51,62
17	51,62
16	51,62
15	51,51
14	51,51
13	51,40
12	51,40
11	51,40
10	51,29
9	51,29
8	51,17
7	51,17
6	51,17
5	51,06
4	51,06
3	50,95
2	50,95

Lampiran 15 : Data Tegangan dan SOC ketika *Charging* pada Baterai DPV

SOC (%)	Tegangan (V)
2	51,4
3	52,36
4	52,68
5	52,68
6	52,84
7	53
8	52,84
9	52,84
10	52,68
11	53,16
12	53,16
13	53
14	53,48
15	53,64
16	53,48
17	53,64
18	53,96
19	53,8
20	53,8
21	54,44
22	53,96
23	53,96
24	54,12
25	53,96
26	54,12

SOC (%)	Tegangan (V)
27	53,8
28	54,12
29	54,12
30	54,28
31	53,96
32	54,12
33	54,12
34	54,12
35	54,12
36	53,16
37	54,12
38	54,28
39	54,44
40	54,92
41	53,96
42	54,44
43	54,28
44	54,28
45	54,28
46	54,28
47	54,44
48	54,44
49	54,28
50	54,28
51	54,28
52	54,44
53	54,44
54	54,44

SOC (%)	Tegangan (V)
55	54,6
56	54,6
57	54,6
58	54,44
59	54,6
60	54,44
61	54,6
62	54,76
63	54,6
64	54,44
65	54,6
66	53,64
67	54,76
68	54,76
69	54,76
70	54,76
71	54,76
72	54,92
73	54,76
74	54,92
75	54,92
76	54,76
77	55,56
78	54,92
79	54,76
80	54,76
81	54,92
82	54,92

SOC (%)	Tegangan (V)
83	54,76
84	54,92
85	52,84
86	54,92
87	54,92
88	54,76
89	54,92
90	54,92
91	54,92
92	55,08
93	55,08
94	55,08
95	55,08
96	55,08
97	55,24
98	53
99	55,24
100	55,24

RIWAYAT PENULIS

Muhammad Nur Aziz Asfar Afif, lahir di Kabupaten Blitar pada 22 Mei 1998. Anak pertama dari dua bersaudara, bertempat tinggal di Perumahan Griya Permata Asri Wlingi, Kabupaten Blitar, Jawa Timur. Pernah menempuh pendidikan di SDN Babadan 1 Wlingi, SMPN 1 Wlingi, dan SMAN 1 Talun. Saat ini sedang menempuh jenjang pendidikan pada Program Studi Komputer Kontrol di Departemen Teknik Elektro Otomasi, Fakultas Vokasi, Institut Teknologi Sepuluh Nopember Surabaya. Menyukai kegiatan olahraga dan jalan-jalan di alam bebas. Mempunyai cita-cita menjadi seorang sosiopreneur.