**BAB 1**

**PENDAHULUAN**

**1.1 Latar Belakang**

Kerja praktik adalah kegiatan mahasiswa yang dilakukan di suatu perusahaan atau instansi, maupun di lingkungan masyarakat untuk mengaplikasikan ilmu yang diperoleh dan melihat relevansinya di perusahaan maupun melalui jalur pengembangan diri dengan mendalam1i bidang ilmu tertentu dan aplikasinya. Kerja praktik merupakan salah satu kurikulum wajib yang harus ditempuh oleh mahasiswa Politeknik Elektronika Negeri Surabaya (PENS). Dari kerja praktik tersebut diharapkan dapat menambah ilmu pengetahuan, menerapkan ilmu pengetahuan yang telah diperoleh selama kuliah, dan dapat menjadikan mahasiswa mampu menghadapi perkembangan teknologi di era globalisasi.

BPPT merupakan lembaga pemerintah non departemen yang dibentuk untuk memberikan saran dan masukan kepada Presiden, sangat dituntut mempunyai dan memahami permasalahan yang terkait dengan kebijakan pemanfaatan teknologi unggul guna membantu mengarahkan kemajuan bangsa dan negara. Perusahaan ini memiliki banyak elemen penting sebagai penyusunnya baik dari segi manajerial maupun praktisi. Dari sisi manajerial terdapat struktur organisasi yang mencakupi peranan fungsi dari seluruh bidang pekerjaan. Jenis bidang pekerjaan di perusahaan ini sangat banyak sehingga menjadi alasan kami mahasiswa Politeknik Elektronika Negeri Surabaya (PENS) untuk melakukan kerja praktik agar kami memperoleh ilmu baik dari segi manajerial maupun praktisi di perusahaan tersebut.

Secara spesifik, Pusat Teknologi Elektronika mempunyai arahan kerja pengkajian dan penerapan dalam bidang Elektronika. Karena bidang tersebut terdapat kecocokan dengan bidang prodi Teknik Elektronika dan kami berkeinginan untuk mempelajari lebih lanjut mengenai Elektronika Optik, maka kami ditempatkan pada Laboratorium Fotonika pada BPPT Kota Tangerang Selatan mengenai VLC (Visible Light Communication).

**1.2** **Perumusan Masalah**

Dengan pelaksanaan kerja praktik ini, terdapat beberapa rumusan masalah yang dibahas :

1. Mempelajari tentang *Visible Light Communication* (VLC).
2. Mempelajari tentang *Internet of Things* (IoT).
3. Mempelajari tentang website developer.
4. Mengerjakan tugas yang diberikan oleh BPPT Lab. Fotonika Kota Tangerang Selatan.
   1. **Tujuan dan Manfaat**
      1. Tujuan

Tujuan dari kerja praktik ini adalah:

1. Untuk memenuhi Satuan Kredit Semester (SKS) yang harus ditempuh sebagai persyaratan akademik jurusan D4 Teknik Elektronika PENS.
2. Mengetahui implementasi dari ilmu yang telah diberikan saat perkuliahan.
3. Menerapkan ilmu yang telah dipelajari selama kuliah ke dalam dunia kerja.
4. Memahami sistem kerja di dunia perusahaan dan mengetahui secara nyata kondisi yang terjadi selama proses riset berlangsung.
   * 1. Manfaat

Manfaat dari kerja praktik ini adalah:

1. Bagi Mahasiswa

* Menambah ilmu pengetahuan mengenai dunia industri.
* Menerapkan ilmu yang telah diperoleh ke dalam dunia industri.
* Melatih skill mahasiswa dalam bidang industri.
* Memperoleh bekal untuk menghadapi permasalahan pada dunia industri.

1. Bagi PENS

* Sarana pengenalan IPTEK khususnya Teknik Elektronika dan sebagai bahan pertimbangan dalam penyusunan program di PENS.
* Sebagai bahan masukan dan evaluasi program pendidikan di PENS untuk menghasilkan tenaga-tenaga yang terampil sesuai dengan kebutuhan industri.

1. Bagi BPPT Serpong, Tangerang Selatan-Banten

* Sarana mengetahui kualitas pendidikan di perguruan tinggi negeri, khususnya PENS.
* Sarana untuk mengenalkan teknologi industri pada dunia pendidikan.
* Sarana untuk memberikan kriteria tenaga kerja yang dibutuhkan oleh badan usaha terkait.
  1. **Ruang Lingkup Pembahasan** 
     1. Metode Tanya Jawab

Pengumpulan data yang dilakukan dengan melakukan tanya jawab secara langsung kepada karyawan agar memperoleh penjelasan mengenai data-data yang diinginkan.

* + 1. Metode Observasi

Pengumpulan data yang dilakukan dengan melakukan pengamatan secara langsung terhadap suatu objek untuk memperoleh data atau informasi kemudian dicatat dengan singkat dan jelas agar bersifat konkrit.

* + 1. Metode Literature

Pengumpulan data yang dilakukan dengan cara membaca dari internet dan dokumen-dokumen industri yang berkaitan dengan objek yang di observasi sehingga memperoleh data yang lengkap.

* 1. **Sistematika Penulisan**

Laporan kerja praktik ini terbagi menjadi 4 bab dimana masing-masing bab terdiri dari beberapa sub bab yang menjelaskan isi dari bab tersebut. Adapun sistematika penulisan laporan ini adalah sebagai berikut:

1. BAB I: Pendahuluan

Menjelaskan tentang hal-hal umum mengenai kerja praktik dan permasalahan yang muncul di tempat kerja praktik. Berisi latar belakang, perumusan masalah, tujuan dan manfaat, metode pengumpulan data, dan sistematika penulisan.

1. BAB II: Gambaran Umum Perusahaan

Menjelaskan tentang gambaran/profil singkat dari perusahaan BPPT kota Tangerang Selatan. Pada bab ini terdiri dari sejarah singkat, struktur organisasi, hak dan wewenang, lokasi perusahaan, dan etika profesi.

1. BAB III: Hasil Kegiatan Praktik

Menjelaskan tentang *Visible Light Communication* (VLC) dan *Internet of Things* (IoT). Pada bab ini terdiri dari proyek yang telah diberikan oleh pembimbing Lab. Fotonika – Pusat Teknologi Elektronika, BPPT Serpong.

1. BAB IV: Penutup

Pada bab ini membahas tentang kesimpulan dari proyek yang telah diberikan oleh pembimbing Lab. Fotonika – Pusat Teknologi Elektronika, BPPT Serpong. dan kolerasi kegiatan KP dengan mata kuliah yang diajarkan serta berisi saran yang diharapkan dapat bermanfaat bagi pembaca.

**BAB 2**

**GAMBARAN UMUM PERUSAHAAN**

* 1. **Sejarah Singkat BPPT**

Badan Pengkajian dan Penerapan Teknologi (BPPT) adalah Lembaga Pemerintah Non-Kementerian yang berada dibawah koordinasi Kementerian Riset, Teknologi dan Pendidikan Tinggi yang mempunyai tugas melaksanakan tugas pemerintahan di bidang pengkajian dan penerapan teknologi.

Proses pembentukan BPPT bermula dari gagasan Presiden RI ke-2, Soeharto kepada Prof Dr. Ing. B.J. Habibie pada tanggal 28-Januari-1974. Dengan surat keputusan no. 76/M/1974 tanggal 5-Januari-1974, Prof Dr. Ing. B.J. Habibie diangkat sebagai penasehat pemerintah di bidang advance teknologi dan teknologi penerbangan yang bertanggung jawab langsung pada presiden dengan membentuk Divisi Teknologi dan Teknologi Penerbangan (ATTP) Pertamina.

Melalui surat keputusan Dewan Komisaris Pemerintah Pertamina No.04/Kpts/DR/DU/1975 tanggal 1 April 1976, ATTP diubah menjadi Divisi Advance Teknologi Pertamina. Kemudian diubah menjadi Badan Pengkajian dan Penerapan Teknologi melalui Keputusan Presiden Republik Indonesia No.25 tanggal 21 Agustus 1978. Diperbaharui dengan Surat Keputusan Presiden No.47 tahun 1991.

* 1. **Logo, Visi, dan Misi Perusahaan**

1. Logo Perusahaan



Gambar 2.1 Logo Perusahaan

Sumber : bppt.go.id

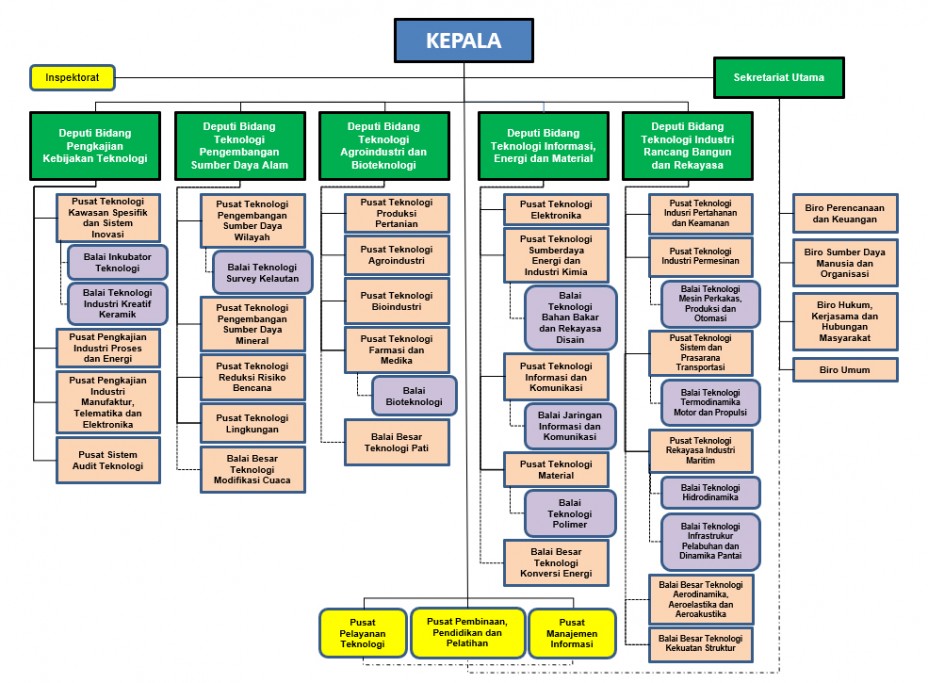
1. Visi Perusahaan

Menjadi lembaga unggulan Teknologi dalam pengkajian dan penerapan teknologi untuk meningkatkan daya saing menuju kemandirian bangsa

1. Misi Perusahaan

Upaya untuk mewujudkan visi BPPT tersebut dilaksanakan melalui 3 (tiga) misi sebagai berikut:

1. Merumuskan dan merekomendasikan kebijakan nasional di bidang teknologi untuk peningkatan daya saing menuju kemandirian bangsa;
2. Melaksanakan pengkajian dan penerapan teknologi untuk menghasilkan inovasi teknologi, audit teknologi, kliring teknologi, alih teknologi, dan layanan teknologi;
3. Melaksanakan tata kelola pemerintahan yang baik melalui reformasi birokrasi.
   1. **Struktur Organisasi BPPT**



**Gambar 2.1** Struktur Organisasi BPPT

Sumber : bppt.go.id

* 1. **Tugas & Fungsi Pusat Teknologi Elektronika**

Tugas :

Melaksanakan pengkajian dan penerapan di bidang teknologi elektronika.

Fungsi :

* Pelaksanaan pengkajian dan penerapan di bidang teknologi elektronika;
* Pelaksanaan pengkajian dan penerapan di bidang teknologi instrumentasi;
* Pelaksanaan pengkajian dan penerapan di bidang teknologi telekomunikasi;
* Penyiapan bahan rumusan kebijakan teknologi elektronika;
* Pelaksanaan perencanaan, monitoring, evaluasi program dan anggaran di lingkungan Pusat Teknologi Elektronika.
  1. **Lokasi Perusahaan**

BPPT mempunyai beberapa lokasi yang terdiri dari Humas, & kawasan Puspiptek Jalan Kawasan Puspiptek, Muncul, Setu. Untuk Bagian Humas berlokasi di Jl. M.H Thamrin No.8 Jakarta Pusat, DKI Jakarta – 10340. Dengan nomor telepon : (021)-3169534 atau Faximile (021)-39838729. E-mail : humas@bppt.go.id.

Pada kerja praktek ini, kami ditempatkan di Laboratorium Fotonika di bawah naungan Pusat Teknologi Elektronika. Untuk Kerja praktik kami dilaksanakan di Gedung Teknologi 3 Lt.3 PUSPITEK, Serpong, Tangerang Selatan 15314

**BAB 3**

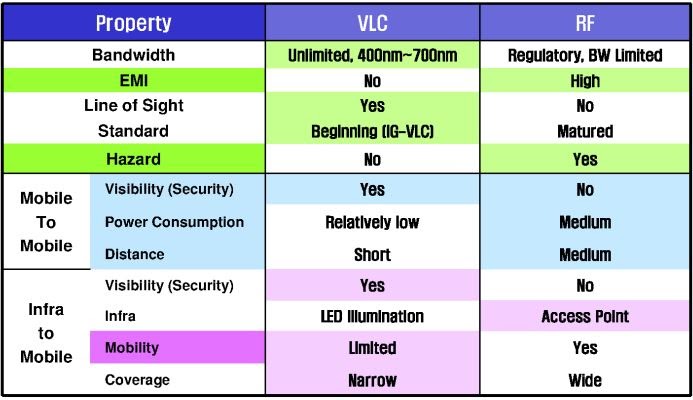
**HASIL KEGIATAN KERJA PRAKTIK**

* 1. **Visible Light Communication (VLC)**

Komunikasi cahaya tampak atau Visible light communication (VLC) adalah sistem komunikasi untuk pengiriman dan penerimaan informasi/data menggunakan gelombang elektromagnetik pada spektrum cahaya tampak antara 780 nm - 375 nm (400 dan 800 THz). Komunikasi ini merupakan salah satu jenis komunikasi nirkabel optik selain komunikasi Radio dan inframerah (IR). Penggunaan cahaya tampak sebagai medium komunikasi menawarkan beberapa keunggulan, yakni salah satunya adalah tidak berbahaya bagi kesehatan manusia, murah dan mudah dalam implementasinya karena infrastruktur telah tersedia yakni perangkat pencahayaan.

VLC adalah teknologi yang muncul yang bermaksud untuk memungkinkan akses internet kecepatan tinggi terutama di lingkungan dalam ruangan dan bekerja berdasarkan prinsip modulasi intensitas infrastruktur pencahayaan keadaan padat yang ada yang sering disediakan oleh Light-Emitting Diode (LED). VLC menawarkan beberapa keuntungan utama dibandingkan jaringan akses tradisional berbasis frekuensi radio termasuk sekitar 300 THz bandwidth bebas lisensi yang dibawa pada panjang gelombang yang terlihat, ~ 10.000 'lebih besar dari yang tersedia di radio, yang juga secara substansial kelebihan permintaan. Menurut perkiraan terbaru, lalu lintas data seluler diperkirakan akan tumbuh pada tingkat pertumbuhan tahunan gabungan sebesar 47% antara 2016-2021. Mempertimbangkan sifat spektrum radio yang sangat padat, solusi pelengkap jelas diperlukan, dan VLC adalah salah satu kandidat utama untuk menyediakan spektrum tambahan.

Pada era revolusi industri 4.0, penggunaan perangkat berbasis *Internet of Things* (IoT) dipastikan akan meningkat pesat. Tentunya hal ini menyebabkan melonjaknya permintaan bandwidth untuk akses saluran komunikasi tiap perangkat IoT. Sementara di satu sisi, bandwidth dari *Radio Frecuency* (RF) lisensi sudah semakin padat dan mulai saling bertumpukan. Tumpukan frekuensi ini akan menyebabkan peningkatan daya elektromagnetik yang pada tingkat tertentu akan berdampak pada lingkungan sekitarnya bahkan pada kesehatan manusia di lingkungan tersebut. VLC hadir sebagai komplementer dari kurangnya ketersediaan frekuensi radio tersebut.Berikut perbandingan antara VLC dengan RF sebagai sarana komunikasi.



Gambar 3.1 Perbandingan antara VLC dengan RF sebagai media komunikasi

* 1. **STM32**

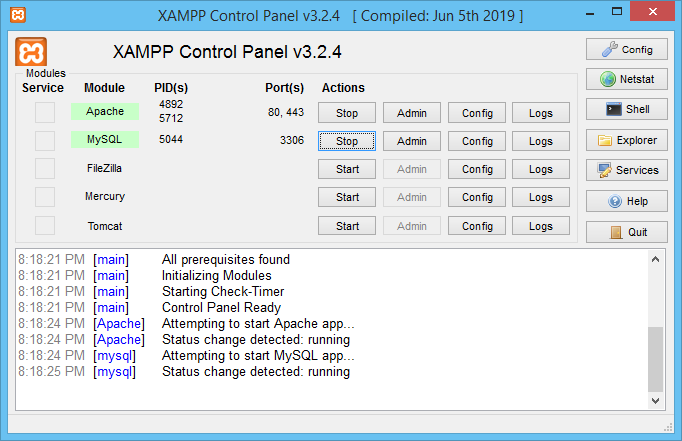
STM32 adalah rangkaian sirkuit terintegrasi mikrokontroler 32-bit oleh STMicroelectronics. Secara internal, setiap mikrokontroler terdiri dari inti prosesor, RAM statis , memori flash , antarmuka debugging, dan berbagai peripheral. STM32 merupakan keluarga IC mikrokontroler yang berbasis pada RISC ARM 32-bit Cortex-M33F, Cortex-M7F, Cortex-M4F, Cortex-M3, Cortex-M0 +, dan Cortex-M0 core. STMicroelectronics melisensikan IP Processor ARM dari ARM Holdings. Desain inti ARM memiliki banyak opsi yang dapat dikonfigurasi, dan ST memilih konfigurasi individual yang akan digunakan untuk setiap desain. ST menempelkan periferal mereka sendiri ke inti sebelum mengubah desain menjadi cetakan silikon.

Tabel berikut meringkas keluarga mikrokontroler STM32

|  |  |
| --- | --- |
| **Seri STM32** | **ARM CPU Core** |
| L5 | Cortex-M33F |
| F7, H7 | Cortex-M7F |
| F3, F4, G4, L4, L4 +, J | Cortex-M4F |
| F1, F2, L1, W, J | Cortex-M3 |
| G0, L0, J | Cortex-M0 + |
| F0, J | Cortex-M0 |

* 1. **XAMPP**

XAMPP merupakan singkatan dari X (sistem operasi apa saja), Apache, MariaDB, PHP dan Perl. Program ini merupakan web server yang mudah digunakan serta dapat melayani tampilan halaman web yang dinamis. Fungsinya adalah sebagai server yang berdiri sendiri (*localhost*), yang terdiri atas program Apache *Server*, MariaDB *database*, dan penerjemah bahasa yang ditulis dengan bahasa pemrograman PHP dan Perl. Pada XAMPP yang kita jalankan hanya dua *service* yaitu Apache dan Mysql, dapat dilihat pada Gambar 2.



Gambar 3.2 Aplikasi XAMPP

* 1. **Visual Studio Code**

Visual Studio Code adalah sumber kode editor yang dikembangkan oleh Microsoft untuk Windows, Linux dan macOS. Hal ini termasuk dukungan untuk debugging, kontrol Git yang tertanam dan GitHub, penyorotan sintaksis, penyelesaian kode cerdas, snippet, dan refactoring kode. Ini memungkinkan pengguna untuk mengubah tema, pintasan keyboard, preferensi, dan menginstal ekstensi yang menambah fungsionalitas tambahan. Kode sumber adalah sumber bebas dan terbuka dan dirilis di bawah Lisensi MIT permisif. Binari yang dikompilasi adalah freeware dan gratis untuk penggunaan pribadi atau komersial. Aplikasi Visual Studio Code dapat dilihat pada Gambar 3.

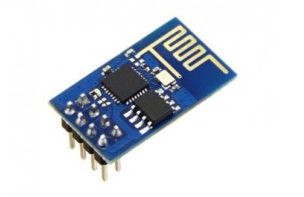
A screenshot of a computer screen

Description automatically generated

Gambar 3.3 Aplikasi Visual Studio Code

* 1. **ESP8266**

ESP8266 digunakan untuk komunikasi data melalui jaringan yang berbasis *Wireless Fidelity* (WIFI). ESP8266 pada alat ini bertugas untuk mengirimkan data rekaman saat mobil memasuki gerbang tol otomatis. ESP8266 dihubungkan ke pinout 0 dan 1 yaitu transmitter dan receiver untuk dapat bisa berinteraksi dengan website yang dibuat. Modul ESP8266 dapat dilihat pada Gambar 1.



Gambar 3.4 Modul ESP8266

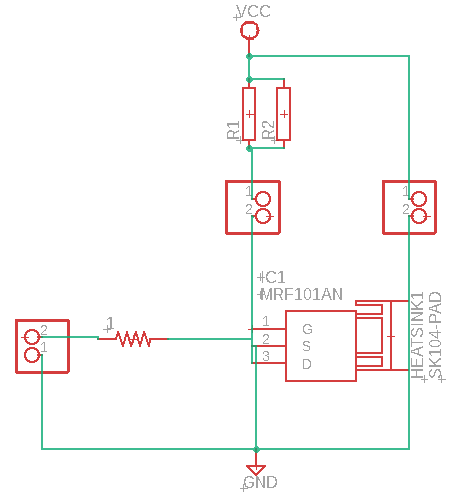
**3. 1 Analisis Masalah**

**3. 2 Analisis Kebutuhan**

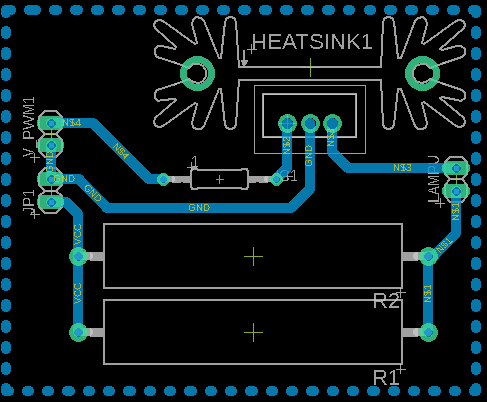
Analisis kebutuhan melingkupi apa saja yang dibutuhkan untuk membuat modulasi *On-Off Keying* (OOK) setelah berdiskusi dengan pihak instansi BPPT. Kebutuhan membuat modulasi OOK terdapat dua macam yaitu pada bagian transmitter dan receiver. Transmitter membutuhkan komponen perangkat keras yaitu STM32F746-NUCLEO, LED 12 watt, dan MRF101AN sedangkan pada receiver membutuhkan Arduino Uno, photodiode dan OPA380AIDGKT. Kebutuhan perangkat lunak yaitu software Arduino dan STM32CUBE IDE.

3.2.1 Rangkaian Transmitter dan Receiver

a. Skema Rangkaian Transmitter



b. Board Rangkaian Transmitter



Selain rangkaian transmitter juga terdapat rangkaian receiver. Dalam rangkaian receiver terdapat rangkaian komparator.

c. Skema Rangkaian Receiver

d. Board Rangkaian Receiver

3.2.2 Hasil Percobaan

Berdasarkan percobaan yang kami lakukan dengan menggunakan teknik modulasi *On-Off Keying* (OOK) maka didapatkan hasil berupa tegangan dan frekuesi pada receiver dan transmitter. Data hasil percobaan dapat dilihat pada table berikut :

1. Hasil pada Receiver

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Bitrate** | **Tegangan Receiver (V)** | | | | **Frekuensi Receiver (kHz)** | | | |
| **30 cm** | **60 cm** | **100 cm** | **150 cm** | **30 cm** | **60 cm** | **100 cm** | **150 cm** |
| **9600** | 4.84 | 4.8 | 3.7 | 3.5 | 47.98 | 4.8 | 4.8 | 4.8 |
| **14400** | 5 | 4.72 | 3.49 | 3.3 | 48 | 4.79 | 4.8 | 2.4 |
| **19200** | 5.08 | 4.64 | 3.36 | 2.16 | 38.39 | 3.84 | 3.78 | 18.9 |
| **38400** | 5 | 4.72 | 3.28 | 2.16 | 19.21 | 19.21 | 19.2 | 19.21 |
| **57600** | 4.92 | 4.72 | 3.28 | 2.08 | 28.74 | 28.8 | 28.79 | 28.79 |
| **115200** | 4.92 | 4.64 | 3.2 | 2 | 57.54 | 57.54 | 57.55 | 57.54 |
| **230400** | 4.92 | 4.64 | 3.28 | 2.08 | 79.19 | 115.4 | 115.3 | 115.5 |

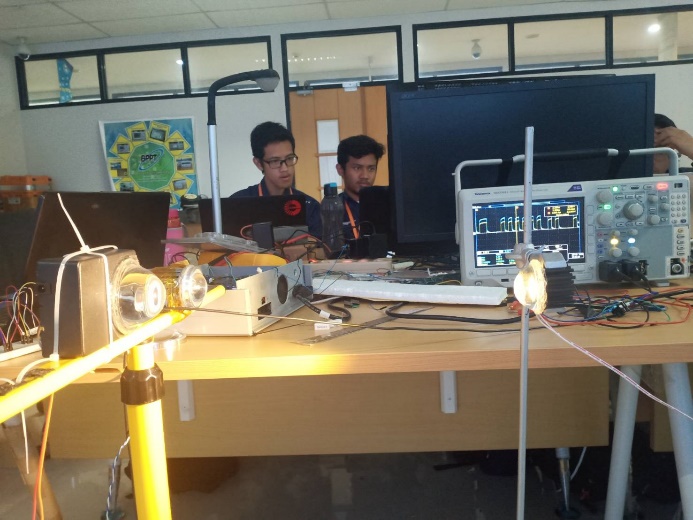
1. Hasil pada Transmitter

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Baudrate** | **Periode Transmitter (us)** | | | | **Tegangan Transmitter (V)** | | | |
| **30 cm** | **60 cm** | **100**  **cm** | **150 cm** | **30 cm** | **60 cm** | **100 cm** | **150 cm** |
| **9600** | 208.4 | 208.4 | 208.4 | 208.4 | 3.44 | 3.44 | 3.44 | 3.44 |
| **14400** | 208.4 | 208.2 | 208.4 | 208.4 | 3.44 | 3.52 | 3.36 | 3.44 |
| **19200** | 260.6 | 260.6 | 264.6 | 104 | 3.44 | 3.36 | 3.44 | 3.36 |
| **38400** | 52.09 | 52.1 | 52.07 | 52.98 | 3.36 | 3.36 | 3.36 | 3.36 |
| **57600** | 34.72 | 34.7 | 34.72 | 34.7 | 3.28 | 3.36 | 3.28 | 3.36 |
| **115200** | 17.37 | 17.38 | 17.37 | 17.38 | 3.2 | 3.2 | 3.28 | 3.2 |
| **230400** | 8.66 | 8.67 | 8.66 | 8.67 | 3.28 | 3.28 | 3.44 | 3.2 |

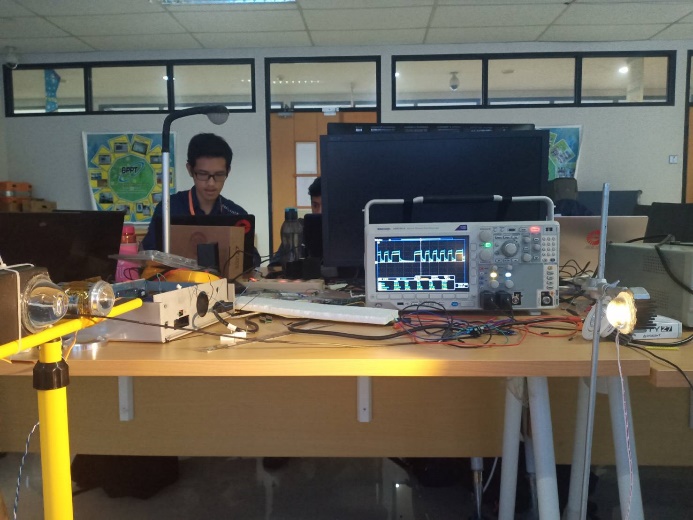
1. Dokumentasi Percobaan

Berikut beberapa dokumentasi dari percobaan yang kami lakukan dengan menggunakan teknik modulasi *On-Off Keying* (OOK)

1. Ketika jarak antara transmitter dan receiver 30 cm



1. Ketika jarak antara transmitter dan receiver 60 cm



1. Ketika jarak antara transmitter dan receiver 100 cm



1. Ketika jarak antara transmitter dan receiver 150 cm



**3. 3 Analisis Pengembangan**

Pada Kerja Praktik dibuat untuk pembuktian bahwa cahaya tampak dapat dimanfaatkan untuk pengiriman data menggunakan modulasi OOK. Sistem Gerbang Toll di Indonesia masih menggunakan tap kartu untuk bisa masuk ke Toll. Hal itu ternyata tidak dapat mengurai kemacetan. Oleh karena itu, kami mengebangkan cahaya tampak untuk diterapkan terhadap gerbang toll yang akan diaplikasikan pada sistem pembayaran tol tanpa adanya tap card dan digantikan oleh sebuah photodiode on board sehingga mobil tidak perlu berhenti pada gerbang tol otomatis tersebut. Dimana ini akan berdampak mengurai kemacetan pada jalan toll.

**3.3.1 Analisis Kebutuhan Pengembangan**

Komponen yang dibutuhkan untuk membuat model alat ini yaitu ESP8266 untuk berkomunikasi antara alat dengan website. Pembuatan website membutuhkan pernagkat lunak seperti XAMPP dan web editor Visual Studio Code untuk membuat web.

**BAB 4**

**KESIMPULAN DAN SARAN**

**4.1 Kesimpulan**

Kerja praktik di Badan Pengkajian dan Penerapan Teknologi memberikan pengalaman yang bermanfaat bagi mahasiswa. Pengenalan terhadap bidang perusahaan merupakan hal pertama yang dilakukan agar mahasiswa mampu mengenali sistem perusahaan dengan baik. Hal tersebut memberikan arah kepada mahasiswa agar mampu menganalisa masalah di dalam perusahaan.

Mahasiswa dituntut untuk mengaplikasikan teori yang selama ini didapatkan dengan dunia penelitian. Salah satu cara untuk mengaplikasikan hal tersebut adalah dengan adanya proyek “Automatic Toll Gate based on VLC”. Proyek ini juga digunakan untuk menambah pengetahuan dan ilmu yang dimiliki. Namun, proyek ini masih terdapat beberapa hal yang belum terealisasikan dikarenakan kurangnya riset dan keterbatasan waktu karena adanya bencana COVID-19 yang mengharuskan Work From Home.

**4.2 Saran**

Dari pelaksanaan kerja praktik selama 3 bulan ini, terdapat beberapa saran dari berbagai pihak. Dengan adanya saran-saran ini, maka diharapkan pelaksanaan kerja praktik dapat lebih baik lagi untuk mahasiswa, kampus, dan perusahaan.

4.2.1 Untuk Mahasiswa

1. Lebih mempersiapkan diri sebelum memulai kerja praktek. Persiapan disini dapat berupa mempelajari bidang perusahaan terlebih dahulu agar mengetahui apa saja yang akan dikerjakan selanjutnya.
2. Lebih berani dalam menyampaikan pendapat atau pertanyaan kepada pembimbing lapangan agar dapat mengetahui lebih lanjut mengenai dunia pekerjaan secara langsung.
3. Mempunyai timeline yang jelas mengenai pengerjaan proyek yang diberikan perusahaan supaya selesai tepat waktu. Selain itu, harus lebih sering berkonsultasi kepada pembimbing lapangan dan dosen pembimbing di kampus.

4.2.2 Untuk Politeknik Elektronika Negeri Surabaya (PENS)

1. Melakukan pemantauan secara berkala kepada mahasiswa kerja praktik sesuai dengan yang disosialisasikan, yaitu dosen mendatangi tempat kerja praktik dari mahasiswa. Hal tersebut guna membangun hubungan baik antara perusahaan dan kampus.
2. Melakukan sosisalisasi beberapa bulan sebelum diadakannya kerja praktik sehingga mahasiswa bisa tahu apa yang akan dilakukan selanjutnya.

4.2.3 Untuk Badan Pengkajian dan Penerapan Teknologi

1. Selalu menyediakan alat atau komponen yang lengkap dan memadai guna mempermudah dalam proses penelitian yang berlangsung di setiap laboratorium.
2. Mengharapkan setiap gedung yang ada di BPPT Serpong, khususnya gedung teknologi 3 pusat teknologi elektronika (PTE) mempunyai kantin terdekat guna memudahkan para pegawai.

**DAFTAR PUSTAKA**

https://pte.bppt.go.id/tentang-kami/tugas-dan-fungsi

<https://id.wikipedia.org/wiki/Komunikasi_cahaya_tampak>

**Halaman ini sengaja dikosongkan**

**LAMPIRAN**

**Lampiran 1. Biodata Penulis**

Penulis 1

|  |  |  |
| --- | --- | --- |
| **BIODATA DIRI** | | |
| Nama | Ilfat Izzat Pratama |  |
| Tempat Lahir | Sampang |
| Tanggal Lahir | 23 Desember 1998 |
| Jenis Kelamin | Laki – Laki |
| Kewarganegaraan | Indonesia |
| Agama | Islam |
| Pekerjaan | Mahasiswa |
| Alamat Asal | Jalan Barisan Indah I No.7, Sampang, Kabupaten Sampang, Madura, Jawa Timur | |
| Alamat Surabaya | Jalan Kejawan Gebang I No.15, Sukolilo, Surabaya, Jawa Timur | |
| Telepon | 082245093894 | |
| E-mail | [*pratama.izzat231298@gmail.com*](mailto:pratama.izzat231298@gmail.com) | |

Penulis 2

|  |  |  |
| --- | --- | --- |
| **BIODATA DIRI** | | |
| Nama | Fithrotul Irda Amaliah | 8446 3x4 |
| Tempat Lahir | Surabaya |
| Tanggal Lahir | 27 Juni 2000 |
| Jenis Kelamin | Perempuan |
| Kewarganegaraan | Indonesia |
| Agama | Islam |
| Pekerjaan | Mahasiswa |
| Alamat Asal | Jl. Pacar Keling V No. 20, Tambaksari, Surabaya, Jawa Timur | |
| Alamat Surabaya | Jl. Pacar Keling V No. 20, Tambaksari, Surabaya, Jawa Timur | |
| Telepon | 081357216987 | |
| E-mail | [*amalia.irda@gmail.com*](mailto:amalia.irda@gmail.com) | |

Penulis 3

|  |  |  |
| --- | --- | --- |
| **BIODATA DIRI** | | |
| Nama | Dita Anggraini Pratiwi |  |
| Tempat Lahir | Sidoarjo |
| Tanggal Lahir | 20 Februari 1999 |
| Jenis Kelamin | Perempuan |
| Kewarganegaraan | Indonesia |
| Agama | Islam |
| Pekerjaan | Mahasiswa |
| Alamat Asal | Sepanjang Tani RT 09 RW 06 Gg. Mujahiddin Jl. Sriti, Taman, Sidoarjo, Jawa Timur | |
| Alamat Surabaya | - | |
| Telepon | 085235831813 | |
| E-mail | [*ditaanggrainipratiwi@gmail.com*](mailto:ditaanggrainipratiwi@gmail.com) | |

**Lampiran 2. Program Project**

**Program Receiver menggunakan Arduino**

char received\_data; //variable to store read data

void setup() {

Serial.begin(9600);

}

void loop() {

if(Serial.available()>0) //check for any data received

{

received\_data = Serial.read(); //read received data

Serial.print("received data is: ");

Serial.println(received\_data); //display received data

}

}

**Program Transmitter menggunakan STM32**

/\* USER CODE BEGIN Header \*/

/\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* @file : main.c

\* @brief : Main program body

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* @attention

\*

\* <h2><center>&copy; Copyright (c) 2020 STMicroelectronics.

\* All rights reserved.</center></h2>

\*

\* This software component is licensed by ST under BSD 3-Clause license,

\* the "License"; You may not use this file except in compliance with the

\* License. You may obtain a copy of the License at:

\* opensource.org/licenses/BSD-3-Clause

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*/

/\* USER CODE END Header \*/

/\* Includes ------------------------------------------------------------------\*/

**#include** "main.h"

/\* Private includes ----------------------------------------------------------\*/

/\* USER CODE BEGIN Includes \*/

**#include** "string.h"

**#include** "stdio.h"

//#include <curl.h>

/\* USER CODE END Includes \*/

/\* Private typedef -----------------------------------------------------------\*/

/\* USER CODE BEGIN PTD \*/

/\* USER CODE END PTD \*/

/\* Private define ------------------------------------------------------------\*/

/\* USER CODE BEGIN PD \*/

/\* USER CODE END PD \*/

/\* Private macro -------------------------------------------------------------\*/

/\* USER CODE BEGIN PM \*/

/\* USER CODE END PM \*/

/\* Private variables ---------------------------------------------------------\*/

ETH\_HandleTypeDef heth;

TIM\_HandleTypeDef htim1;

TIM\_HandleTypeDef htim3;

UART\_HandleTypeDef huart4;

UART\_HandleTypeDef huart3;

PCD\_HandleTypeDef hpcd\_USB\_OTG\_FS;

/\* USER CODE BEGIN PV \*/

**unsigned** **int** PWM\_Speed = 3;

uint16\_t data\_buffer[9];

uint16\_t pwm1\_termodulasi[9];

uint16\_t pwm0,pwm1;

uint16\_t indeks, start, stop, space;

uint16\_t i =0;

/\* USER CODE END PV \*/

/\* Private function prototypes -----------------------------------------------\*/

**void** **SystemClock\_Config**(**void**);

**static** **void** **MX\_GPIO\_Init**(**void**);

**static** **void** **MX\_ETH\_Init**(**void**);

**static** **void** **MX\_USART3\_UART\_Init**(**void**);

**static** **void** **MX\_USB\_OTG\_FS\_PCD\_Init**(**void**);

**static** **void** **MX\_TIM3\_Init**(**void**);

**static** **void** **MX\_UART4\_Init**(**void**);

**static** **void** **MX\_TIM1\_Init**(**void**);

/\* USER CODE BEGIN PFP \*/

**void** **UART\_Print**(**char** \*);

**void** **UART\_Println**( **char** \*);

**void** **UART\_PrintNum**(uint16\_t);

**void** **UART\_PrintNumln**(uint16\_t);

**void** **modulasi\_1PWM**(**char**, **int**);

uint32\_t **fungsi\_1pwm**(uint16\_t);

/\* USER CODE END PFP \*/

/\* Private user code ---------------------------------------------------------\*/

/\* USER CODE BEGIN 0 \*/

/\* USER CODE END 0 \*/

/\*\*

\* @brief The application entry point.

\* @retval int

\*/

**int** **main**(**void**)

{

/\* USER CODE BEGIN 1 \*/

/\* USER CODE END 1 \*/

/\* MCU Configuration--------------------------------------------------------\*/

/\* Reset of all peripherals, Initializes the Flash interface and the Systick. \*/

HAL\_Init();

/\* USER CODE BEGIN Init \*/

/\* USER CODE END Init \*/

/\* Configure the system clock \*/

SystemClock\_Config();

/\* USER CODE BEGIN SysInit \*/

/\* USER CODE END SysInit \*/

/\* Initialize all configured peripherals \*/

MX\_GPIO\_Init();

MX\_ETH\_Init();

MX\_USART3\_UART\_Init();

MX\_USB\_OTG\_FS\_PCD\_Init();

MX\_TIM3\_Init();

MX\_UART4\_Init();

MX\_TIM1\_Init();

/\* USER CODE BEGIN 2 \*/

//HAL\_TIM\_PWM\_Start(&htim3, TIM\_CHANNEL\_1);

//htim3.Instance->CCR1=800;

/\* USER CODE END 2 \*/

/\* Infinite loop \*/

/\* USER CODE BEGIN WHILE \*/

**while** (1)

{

//modulasi\_1PWM('U',3);

UART\_Print("G");

// HAL\_GPIO\_TogglePin(Out\_GPIO\_Port, Out\_Pin);

HAL\_Delay(1);

// UART\_PrintNumln(123);

/\* USER CODE END WHILE \*/

/\* USER CODE BEGIN 3 \*/

}

/\* USER CODE END 3 \*/

}

/\*\*

\* @brief System Clock Configuration

\* @retval None

\*/

**void** **SystemClock\_Config**(**void**)

{

RCC\_OscInitTypeDef RCC\_OscInitStruct = {0};

RCC\_ClkInitTypeDef RCC\_ClkInitStruct = {0};

RCC\_PeriphCLKInitTypeDef PeriphClkInitStruct = {0};

/\*\* Configure LSE Drive Capability

\*/

HAL\_PWR\_EnableBkUpAccess();

/\*\* Configure the main internal regulator output voltage

\*/

\_\_HAL\_RCC\_PWR\_CLK\_ENABLE();

\_\_HAL\_PWR\_VOLTAGESCALING\_CONFIG(PWR\_REGULATOR\_VOLTAGE\_SCALE1);

/\*\* Initializes the CPU, AHB and APB busses clocks

\*/

RCC\_OscInitStruct.OscillatorType = RCC\_OSCILLATORTYPE\_HSE;

RCC\_OscInitStruct.HSEState = RCC\_HSE\_ON;

RCC\_OscInitStruct.PLL.PLLState = RCC\_PLL\_ON;

RCC\_OscInitStruct.PLL.PLLSource = RCC\_PLLSOURCE\_HSE;

RCC\_OscInitStruct.PLL.PLLM = 4;

RCC\_OscInitStruct.PLL.PLLN = 216;

RCC\_OscInitStruct.PLL.PLLP = RCC\_PLLP\_DIV2;

RCC\_OscInitStruct.PLL.PLLQ = 9;

**if** (HAL\_RCC\_OscConfig(&RCC\_OscInitStruct) != *HAL\_OK*)

{

Error\_Handler();

}

/\*\* Activate the Over-Drive mode

\*/

**if** (HAL\_PWREx\_EnableOverDrive() != *HAL\_OK*)

{

Error\_Handler();

}

/\*\* Initializes the CPU, AHB and APB busses clocks

\*/

RCC\_ClkInitStruct.ClockType = RCC\_CLOCKTYPE\_HCLK|RCC\_CLOCKTYPE\_SYSCLK

|RCC\_CLOCKTYPE\_PCLK1|RCC\_CLOCKTYPE\_PCLK2;

RCC\_ClkInitStruct.SYSCLKSource = RCC\_SYSCLKSOURCE\_PLLCLK;

RCC\_ClkInitStruct.AHBCLKDivider = RCC\_SYSCLK\_DIV1;

RCC\_ClkInitStruct.APB1CLKDivider = RCC\_HCLK\_DIV4;

RCC\_ClkInitStruct.APB2CLKDivider = RCC\_HCLK\_DIV2;

**if** (HAL\_RCC\_ClockConfig(&RCC\_ClkInitStruct, FLASH\_LATENCY\_7) != *HAL\_OK*)

{

Error\_Handler();

}

PeriphClkInitStruct.PeriphClockSelection = RCC\_PERIPHCLK\_USART3|RCC\_PERIPHCLK\_UART4

|RCC\_PERIPHCLK\_CLK48;

PeriphClkInitStruct.Usart3ClockSelection = RCC\_USART3CLKSOURCE\_PCLK1;

PeriphClkInitStruct.Uart4ClockSelection = RCC\_UART4CLKSOURCE\_PCLK1;

PeriphClkInitStruct.Clk48ClockSelection = RCC\_CLK48SOURCE\_PLL;

**if** (HAL\_RCCEx\_PeriphCLKConfig(&PeriphClkInitStruct) != *HAL\_OK*)

{

Error\_Handler();

}

}

/\*\*

\* @brief ETH Initialization Function

\* @param None

\* @retval None

\*/

**static** **void** **MX\_ETH\_Init**(**void**)

{

/\* USER CODE BEGIN ETH\_Init 0 \*/

/\* USER CODE END ETH\_Init 0 \*/

uint8\_t MACAddr[6] ;

/\* USER CODE BEGIN ETH\_Init 1 \*/

/\* USER CODE END ETH\_Init 1 \*/

heth.Instance = ETH;

heth.Init.AutoNegotiation = ETH\_AUTONEGOTIATION\_ENABLE;

heth.Init.PhyAddress = LAN8742A\_PHY\_ADDRESS;

MACAddr[0] = 0x00;

MACAddr[1] = 0x80;

MACAddr[2] = 0xE1;

MACAddr[3] = 0x00;

MACAddr[4] = 0x00;

MACAddr[5] = 0x00;

heth.Init.MACAddr = &MACAddr[0];

heth.Init.RxMode = ETH\_RXPOLLING\_MODE;

heth.Init.ChecksumMode = ETH\_CHECKSUM\_BY\_HARDWARE;

heth.Init.MediaInterface = ETH\_MEDIA\_INTERFACE\_RMII;

/\* USER CODE BEGIN MACADDRESS \*/

/\* USER CODE END MACADDRESS \*/

**if** (HAL\_ETH\_Init(&heth) != *HAL\_OK*)

{

Error\_Handler();

}

/\* USER CODE BEGIN ETH\_Init 2 \*/

/\* USER CODE END ETH\_Init 2 \*/

}

/\*\*

\* @brief TIM1 Initialization Function

\* @param None

\* @retval None

\*/

**static** **void** **MX\_TIM1\_Init**(**void**)

{

/\* USER CODE BEGIN TIM1\_Init 0 \*/

/\* USER CODE END TIM1\_Init 0 \*/

TIM\_ClockConfigTypeDef sClockSourceConfig = {0};

TIM\_MasterConfigTypeDef sMasterConfig = {0};

TIM\_OC\_InitTypeDef sConfigOC = {0};

TIM\_BreakDeadTimeConfigTypeDef sBreakDeadTimeConfig = {0};

/\* USER CODE BEGIN TIM1\_Init 1 \*/

/\* USER CODE END TIM1\_Init 1 \*/

htim1.Instance = TIM1;

htim1.Init.Prescaler = 0;

htim1.Init.CounterMode = TIM\_COUNTERMODE\_UP;

htim1.Init.Period = 0;

htim1.Init.ClockDivision = TIM\_CLOCKDIVISION\_DIV1;

htim1.Init.RepetitionCounter = 0;

htim1.Init.AutoReloadPreload = TIM\_AUTORELOAD\_PRELOAD\_DISABLE;

**if** (HAL\_TIM\_Base\_Init(&htim1) != *HAL\_OK*)

{

Error\_Handler();

}

sClockSourceConfig.ClockSource = TIM\_CLOCKSOURCE\_INTERNAL;

**if** (HAL\_TIM\_ConfigClockSource(&htim1, &sClockSourceConfig) != *HAL\_OK*)

{

Error\_Handler();

}

**if** (HAL\_TIM\_PWM\_Init(&htim1) != *HAL\_OK*)

{

Error\_Handler();

}

sMasterConfig.MasterOutputTrigger = TIM\_TRGO\_RESET;

sMasterConfig.MasterOutputTrigger2 = TIM\_TRGO2\_RESET;

sMasterConfig.MasterSlaveMode = TIM\_MASTERSLAVEMODE\_DISABLE;

**if** (HAL\_TIMEx\_MasterConfigSynchronization(&htim1, &sMasterConfig) != *HAL\_OK*)

{

Error\_Handler();

}

sConfigOC.OCMode = TIM\_OCMODE\_PWM1;

sConfigOC.Pulse = 0;

sConfigOC.OCPolarity = TIM\_OCPOLARITY\_HIGH;

sConfigOC.OCNPolarity = TIM\_OCNPOLARITY\_HIGH;

sConfigOC.OCFastMode = TIM\_OCFAST\_DISABLE;

sConfigOC.OCIdleState = TIM\_OCIDLESTATE\_RESET;

sConfigOC.OCNIdleState = TIM\_OCNIDLESTATE\_RESET;

**if** (HAL\_TIM\_PWM\_ConfigChannel(&htim1, &sConfigOC, TIM\_CHANNEL\_1) != *HAL\_OK*)

{

Error\_Handler();

}

sBreakDeadTimeConfig.OffStateRunMode = TIM\_OSSR\_DISABLE;

sBreakDeadTimeConfig.OffStateIDLEMode = TIM\_OSSI\_DISABLE;

sBreakDeadTimeConfig.LockLevel = TIM\_LOCKLEVEL\_OFF;

sBreakDeadTimeConfig.DeadTime = 0;

sBreakDeadTimeConfig.BreakState = TIM\_BREAK\_DISABLE;

sBreakDeadTimeConfig.BreakPolarity = TIM\_BREAKPOLARITY\_HIGH;

sBreakDeadTimeConfig.BreakFilter = 0;

sBreakDeadTimeConfig.Break2State = TIM\_BREAK2\_DISABLE;

sBreakDeadTimeConfig.Break2Polarity = TIM\_BREAK2POLARITY\_HIGH;

sBreakDeadTimeConfig.Break2Filter = 0;

sBreakDeadTimeConfig.AutomaticOutput = TIM\_AUTOMATICOUTPUT\_DISABLE;

**if** (HAL\_TIMEx\_ConfigBreakDeadTime(&htim1, &sBreakDeadTimeConfig) != *HAL\_OK*)

{

Error\_Handler();

}

/\* USER CODE BEGIN TIM1\_Init 2 \*/

/\* USER CODE END TIM1\_Init 2 \*/

HAL\_TIM\_MspPostInit(&htim1);

}

/\*\*

\* @brief TIM3 Initialization Function

\* @param None

\* @retval None

\*/

**static** **void** **MX\_TIM3\_Init**(**void**)

{

/\* USER CODE BEGIN TIM3\_Init 0 \*/

/\* USER CODE END TIM3\_Init 0 \*/

TIM\_ClockConfigTypeDef sClockSourceConfig = {0};

TIM\_MasterConfigTypeDef sMasterConfig = {0};

TIM\_OC\_InitTypeDef sConfigOC = {0};

/\* USER CODE BEGIN TIM3\_Init 1 \*/

/\* USER CODE END TIM3\_Init 1 \*/

htim3.Instance = TIM3;

htim3.Init.Prescaler = 216-1;

htim3.Init.CounterMode = TIM\_COUNTERMODE\_UP;

htim3.Init.Period = 1000-1;

htim3.Init.ClockDivision = TIM\_CLOCKDIVISION\_DIV1;

htim3.Init.AutoReloadPreload = TIM\_AUTORELOAD\_PRELOAD\_DISABLE;

**if** (HAL\_TIM\_Base\_Init(&htim3) != *HAL\_OK*)

{

Error\_Handler();

}

sClockSourceConfig.ClockSource = TIM\_CLOCKSOURCE\_INTERNAL;

**if** (HAL\_TIM\_ConfigClockSource(&htim3, &sClockSourceConfig) != *HAL\_OK*)

{

Error\_Handler();

}

**if** (HAL\_TIM\_PWM\_Init(&htim3) != *HAL\_OK*)

{

Error\_Handler();

}

sMasterConfig.MasterOutputTrigger = TIM\_TRGO\_RESET;

sMasterConfig.MasterSlaveMode = TIM\_MASTERSLAVEMODE\_DISABLE;

**if** (HAL\_TIMEx\_MasterConfigSynchronization(&htim3, &sMasterConfig) != *HAL\_OK*)

{

Error\_Handler();

}

sConfigOC.OCMode = TIM\_OCMODE\_PWM1;

sConfigOC.Pulse = 0;

sConfigOC.OCPolarity = TIM\_OCPOLARITY\_HIGH;

sConfigOC.OCFastMode = TIM\_OCFAST\_DISABLE;

**if** (HAL\_TIM\_PWM\_ConfigChannel(&htim3, &sConfigOC, TIM\_CHANNEL\_1) != *HAL\_OK*)

{

Error\_Handler();

}

/\* USER CODE BEGIN TIM3\_Init 2 \*/

/\* USER CODE END TIM3\_Init 2 \*/

HAL\_TIM\_MspPostInit(&htim3);

}

/\*\*

\* @brief UART4 Initialization Function

\* @param None

\* @retval None

\*/

**static** **void** **MX\_UART4\_Init**(**void**)

{

/\* USER CODE BEGIN UART4\_Init 0 \*/

/\* USER CODE END UART4\_Init 0 \*/

/\* USER CODE BEGIN UART4\_Init 1 \*/

/\* USER CODE END UART4\_Init 1 \*/

huart4.Instance = UART4;

huart4.Init.BaudRate = 9600;

huart4.Init.WordLength = UART\_WORDLENGTH\_8B;

huart4.Init.StopBits = UART\_STOPBITS\_1;

huart4.Init.Parity = UART\_PARITY\_NONE;

huart4.Init.Mode = UART\_MODE\_TX;

huart4.Init.HwFlowCtl = UART\_HWCONTROL\_NONE;

huart4.Init.OverSampling = UART\_OVERSAMPLING\_16;

huart4.Init.OneBitSampling = UART\_ONE\_BIT\_SAMPLE\_DISABLE;

huart4.AdvancedInit.AdvFeatureInit = UART\_ADVFEATURE\_NO\_INIT;

**if** (HAL\_UART\_Init(&huart4) != *HAL\_OK*)

{

Error\_Handler();

}

/\* USER CODE BEGIN UART4\_Init 2 \*/

/\* USER CODE END UART4\_Init 2 \*/

}

/\*\*

\* @brief USART3 Initialization Function

\* @param None

\* @retval None

\*/

**static** **void** **MX\_USART3\_UART\_Init**(**void**)

{

/\* USER CODE BEGIN USART3\_Init 0 \*/

/\* USER CODE END USART3\_Init 0 \*/

/\* USER CODE BEGIN USART3\_Init 1 \*/

/\* USER CODE END USART3\_Init 1 \*/

huart3.Instance = USART3;

huart3.Init.BaudRate = 115200;

huart3.Init.WordLength = UART\_WORDLENGTH\_8B;

huart3.Init.StopBits = UART\_STOPBITS\_1;

huart3.Init.Parity = UART\_PARITY\_NONE;

huart3.Init.Mode = UART\_MODE\_TX\_RX;

huart3.Init.HwFlowCtl = UART\_HWCONTROL\_NONE;

huart3.Init.OverSampling = UART\_OVERSAMPLING\_16;

huart3.Init.OneBitSampling = UART\_ONE\_BIT\_SAMPLE\_DISABLE;

huart3.AdvancedInit.AdvFeatureInit = UART\_ADVFEATURE\_NO\_INIT;

**if** (HAL\_UART\_Init(&huart3) != *HAL\_OK*)

{

Error\_Handler();

}

/\* USER CODE BEGIN USART3\_Init 2 \*/

/\* USER CODE END USART3\_Init 2 \*/

}

/\*\*

\* @brief USB\_OTG\_FS Initialization Function

\* @param None

\* @retval None

\*/

**static** **void** **MX\_USB\_OTG\_FS\_PCD\_Init**(**void**)

{

/\* USER CODE BEGIN USB\_OTG\_FS\_Init 0 \*/

/\* USER CODE END USB\_OTG\_FS\_Init 0 \*/

/\* USER CODE BEGIN USB\_OTG\_FS\_Init 1 \*/

/\* USER CODE END USB\_OTG\_FS\_Init 1 \*/

hpcd\_USB\_OTG\_FS.Instance = USB\_OTG\_FS;

hpcd\_USB\_OTG\_FS.Init.dev\_endpoints = 6;

hpcd\_USB\_OTG\_FS.Init.speed = PCD\_SPEED\_FULL;

hpcd\_USB\_OTG\_FS.Init.dma\_enable = *DISABLE*;

hpcd\_USB\_OTG\_FS.Init.phy\_itface = PCD\_PHY\_EMBEDDED;

hpcd\_USB\_OTG\_FS.Init.Sof\_enable = *ENABLE*;

hpcd\_USB\_OTG\_FS.Init.low\_power\_enable = *DISABLE*;

hpcd\_USB\_OTG\_FS.Init.lpm\_enable = *DISABLE*;

hpcd\_USB\_OTG\_FS.Init.vbus\_sensing\_enable = *ENABLE*;

hpcd\_USB\_OTG\_FS.Init.use\_dedicated\_ep1 = *DISABLE*;

**if** (HAL\_PCD\_Init(&hpcd\_USB\_OTG\_FS) != *HAL\_OK*)

{

Error\_Handler();

}

/\* USER CODE BEGIN USB\_OTG\_FS\_Init 2 \*/

/\* USER CODE END USB\_OTG\_FS\_Init 2 \*/

}

/\*\*

\* @brief GPIO Initialization Function

\* @param None

\* @retval None

\*/

**static** **void** **MX\_GPIO\_Init**(**void**)

{

GPIO\_InitTypeDef GPIO\_InitStruct = {0};

/\* GPIO Ports Clock Enable \*/

\_\_HAL\_RCC\_GPIOC\_CLK\_ENABLE();

\_\_HAL\_RCC\_GPIOH\_CLK\_ENABLE();

\_\_HAL\_RCC\_GPIOA\_CLK\_ENABLE();

\_\_HAL\_RCC\_GPIOB\_CLK\_ENABLE();

\_\_HAL\_RCC\_GPIOE\_CLK\_ENABLE();

\_\_HAL\_RCC\_GPIOD\_CLK\_ENABLE();

\_\_HAL\_RCC\_GPIOG\_CLK\_ENABLE();

/\*Configure GPIO pin Output Level \*/

HAL\_GPIO\_WritePin(GPIOB, LD1\_Pin|LD3\_Pin|LD2\_Pin, *GPIO\_PIN\_RESET*);

/\*Configure GPIO pin Output Level \*/

HAL\_GPIO\_WritePin(Out\_GPIO\_Port, Out\_Pin, *GPIO\_PIN\_RESET*);

/\*Configure GPIO pin Output Level \*/

HAL\_GPIO\_WritePin(USB\_PowerSwitchOn\_GPIO\_Port, USB\_PowerSwitchOn\_Pin, *GPIO\_PIN\_RESET*);

/\*Configure GPIO pin : USER\_Btn\_Pin \*/

GPIO\_InitStruct.Pin = USER\_Btn\_Pin;

GPIO\_InitStruct.Mode = GPIO\_MODE\_IT\_RISING;

GPIO\_InitStruct.Pull = GPIO\_NOPULL;

HAL\_GPIO\_Init(USER\_Btn\_GPIO\_Port, &GPIO\_InitStruct);

/\*Configure GPIO pins : LD1\_Pin LD3\_Pin LD2\_Pin \*/

GPIO\_InitStruct.Pin = LD1\_Pin|LD3\_Pin|LD2\_Pin;

GPIO\_InitStruct.Mode = GPIO\_MODE\_OUTPUT\_PP;

GPIO\_InitStruct.Pull = GPIO\_NOPULL;

GPIO\_InitStruct.Speed = GPIO\_SPEED\_FREQ\_LOW;

HAL\_GPIO\_Init(GPIOB, &GPIO\_InitStruct);

/\*Configure GPIO pin : Out\_Pin \*/

GPIO\_InitStruct.Pin = Out\_Pin;

GPIO\_InitStruct.Mode = GPIO\_MODE\_OUTPUT\_PP;

GPIO\_InitStruct.Pull = GPIO\_NOPULL;

GPIO\_InitStruct.Speed = GPIO\_SPEED\_FREQ\_LOW;

HAL\_GPIO\_Init(Out\_GPIO\_Port, &GPIO\_InitStruct);

/\*Configure GPIO pin : USB\_PowerSwitchOn\_Pin \*/

GPIO\_InitStruct.Pin = USB\_PowerSwitchOn\_Pin;

GPIO\_InitStruct.Mode = GPIO\_MODE\_OUTPUT\_PP;

GPIO\_InitStruct.Pull = GPIO\_NOPULL;

GPIO\_InitStruct.Speed = GPIO\_SPEED\_FREQ\_LOW;

HAL\_GPIO\_Init(USB\_PowerSwitchOn\_GPIO\_Port, &GPIO\_InitStruct);

/\*Configure GPIO pin : USB\_OverCurrent\_Pin \*/

GPIO\_InitStruct.Pin = USB\_OverCurrent\_Pin;

GPIO\_InitStruct.Mode = GPIO\_MODE\_INPUT;

GPIO\_InitStruct.Pull = GPIO\_NOPULL;

HAL\_GPIO\_Init(USB\_OverCurrent\_GPIO\_Port, &GPIO\_InitStruct);

}

/\* USER CODE BEGIN 4 \*/

**void** **UART\_Print**(**char** \*pData)

{

HAL\_UART\_Transmit(&huart4, (uint8\_t \*)pData, **strlen**(pData), 10);

}

**void** **UART\_Println**( **char** \*pData)

{

HAL\_UART\_Transmit(&huart4, (uint8\_t \*)pData, **strlen**(pData), 10);

**char** newLine[2] = "\n\r";

HAL\_UART\_Transmit(&huart4, (uint8\_t \*)newLine, 2, 10);

}

**void** **UART\_PrintNum**(uint16\_t NumVal)

{

**char** buff[100];

**sprintf**(buff, "%d", NumVal);

HAL\_UART\_Transmit(&huart4, (uint8\_t \*)buff, **strlen**(buff), 10);

}

**void** **UART\_PrintNumln**(uint16\_t NumVal)

{

**char** buff[100];

**sprintf**(buff, "%d", NumVal);

HAL\_UART\_Transmit(&huart4, (uint8\_t \*)buff, **strlen**(buff), 10);

**char** newLine[2] = "\n\r";

HAL\_UART\_Transmit(&huart4, (uint8\_t \*)newLine, 2, 10);

}

uint32\_t **fungsi\_1pwm**(uint16\_t pData){

**if**(pData==0)

**return** 700;

**else**

**return** 400;

}

**void** **modulasi\_1PWM**(**char** pData, **int** pwm\_speed)

{

HAL\_TIM\_PWM\_Start(&htim3, TIM\_CHANNEL\_1);

i++;

**if**(pwm\_speed==1){

pwm0 = 170;

pwm1 = 100;

start = 250;

stop = 360;

space = 150;

}

**if**(pwm\_speed==2){

pwm0 = 550;

pwm1 = 320;

start = 780;

stop = 1000;

space = 250;

}

**if**(pwm\_speed==3){

pwm0 = 1500;

pwm1 = 900;

start = 2000;

stop = 2500;

space = 800;

}

data\_buffer[1] = pData & (1<<0);

data\_buffer[2] = pData & (1<<1);

data\_buffer[3] = pData & (1<<2);

data\_buffer[4] = pData & (1<<3);

data\_buffer[5] = pData & (1<<4);

data\_buffer[6] = pData & (1<<5);

data\_buffer[7] = pData & (1<<6);

data\_buffer[8] = pData & (1<<7);

**for**(indeks=1;indeks<9;indeks++){

pwm1\_termodulasi[indeks]=fungsi\_1pwm(data\_buffer[indeks]);

}

htim3.Instance->CCR1=300;

HAL\_Delay(10);

**for**(indeks=1; indeks<9; indeks++){

htim3.Instance->CCR1=pwm1\_termodulasi[indeks];

HAL\_Delay(10);

}

htim3.Instance->CCR1=htim3.Instance->CCR1=900;

HAL\_Delay(10);

}

/\* USER CODE END 4 \*/

/\*\*

\* @brief This function is executed in case of error occurrence.

\* @retval None

\*/

**void** **Error\_Handler**(**void**)

{

/\* USER CODE BEGIN Error\_Handler\_Debug \*/

/\* User can add his own implementation to report the HAL error return state \*/

/\* USER CODE END Error\_Handler\_Debug \*/

}

**#ifdef** USE\_FULL\_ASSERT

/\*\*

\* @brief Reports the name of the source file and the source line number

\* where the assert\_param error has occurred.

\* @param file: pointer to the source file name

\* @param line: assert\_param error line source number

\* @retval None

\*/

**void** assert\_failed(uint8\_t \*file, uint32\_t line)

{

/\* USER CODE BEGIN 6 \*/

/\* User can add his own implementation to report the file name and line number,

tex: printf("Wrong parameters value: file %s on line %d\r\n", file, line) \*/

/\* USER CODE END 6 \*/

}

**#endif** /\* USE\_FULL\_ASSERT \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* (C) COPYRIGHT STMicroelectronics \*\*\*\*\*END OF FILE\*\*\*\*/