BAB II
TINJAUAN PUSTAKA

2.1 Penelitian Relevan

Kajian pustaka terhadap penelitian-penelitian terdahulu merupakan langkah fundamental dalam setiap penelitian ilmiah untuk menghindari duplikasi, memetakan perkembangan *state of the art*, serta mengidentifikasi celah penelitian (*research gap*) yang dapat diisi oleh penelitian yang sedang dilakukan. Tinjauan pustaka ini difokuskan pada penelitian-penelitian yang berkaitan dengan pengembangan sistem diagnostik, *monitoring*, dan modifikasi parameter *Electronic Control Unit* (ECU) sepeda motor injeksi yang dipublikasikan dalam rentang waktu 2020 hingga 2025. Berdasarkan hasil penelusuran literatur, ditemukan sebelas penelitian relevan yang memiliki keterkaitan dengan topik pengembangan sistem diagnostik dan *remapping* ECU sepeda motor. Masing-masing penelitian dianalisis berdasarkan metode yang digunakan, kontribusi yang diberikan, serta keterbatasan yang dapat dijadikan celah untuk pengembangan lebih lanjut. Ringkasan analisis terhadap penelitian-penelitian tersebut disajikan pada Tabel 2. 1.

Tabel 2. 1 Penelitian Terdahulu dan Analisis Celah Penelitian

No	Peneliti & Tahun	Fokus Penelitian	Metode	Analisis Celah (Gap)
1	(Septiandes et al., 2020)	Rancang bangun RPM-Meter sepeda motor injeksi dengan sensor induksi	Sensor induksi elektromagnetik untuk mendeteksi putaran mesin	Pembacaan RPM rentan terhadap noise elektromagnetik dari sistem kelistrikan kendaraan. Data bukan berasal langsung dari ECU sehingga akurasi bergantung pada kualitas sensor.

No	Peneliti & Tahun	Fokus Penelitian	Metode	Analisis Celah (Gap)
2	(Sholeh et al., 2020)	Sistem monitoring kondisi kendaraan motor injeksi berbasis mikrokontroler	Pemasangan sensor-sensor eksternal tambahan pada kendaraan	Tidak efisien karena melakukan duplikasi sensor yang sebenarnya sudah tersedia pada ECU. Biaya implementasi meningkat dan instalasi menjadi kompleks.
3	(Fauzi, 2021)	Perancangan sistem penyalaan engine sepeda motor berbasis Arduino	Arduino dengan komunikasi serial Virtual COM Port (VCP)	Latensi komunikasi tinggi (~16ms) akibat penggunaan driver VCP standar Windows yang tidak optimal untuk protokol dengan timing presisi.
4	(Sari & Wailanduw, 2022)	Rancang bangun alat monitoring kerja sensor pada sepeda motor injeksi	Arduino Nano untuk pembacaan data sensor	Fungsi terbatas pada mode read-only (monitoring saja). Tidak memiliki kemampuan untuk menulis atau memodifikasi parameter ECU (remapping).
5	(Mindarta et al., 2022)	Penerapan diagnostic tool motor injeksi versi Android di bengkel	Aplikasi Android dengan protokol OBD-II standar	Menggunakan protokol OBD-II generik yang tidak dapat mengakses area memori proprietary ECU Honda. Fitur terbatas pada pembacaan DTC standar.

No	Peneliti & Tahun	Fokus Penelitian	Metode	Analisis Celah (Gap)
6	(Fatra et al., 2023)	Analisis re-mapping ECU terhadap performa mesin sepeda motor injeksi 150cc	Penggunaan software remapping komersial (commercial off-the-shelf)	Tidak membangun sistem remapping sendiri, melainkan menggunakan software jadi. Ketergantungan pada vendor dan tidak ada kontrol terhadap algoritma keamanan data.
7	(Damayanti & Septiyanto, 2024)	Pengaruh stroke up dan bore up terhadap performa sepeda motor Honda Verza	Modifikasi fisik mesin (stroke dan bore up) dengan validasi dynotest	Fokus pada modifikasi komponen mekanis mesin, bukan pada aspek elektronik dan protokol komunikasi data ECU.
8	(Pratama et al., 2024)	Analisis performa sepeda motor injeksi 110cc menggunakan ECU standar dan ECU remap	Perbandingan output performa antara ECU standar dengan ECU yang telah di-remap	Fokus pada analisis output performa (daya dan torsi) tanpa membahas aspek teknis proses remapping, keamanan data, atau protokol komunikasi yang digunakan.
9	(Muslim & Subagio, 2025)	Perancangan dan pembuatan interface komunikasi data antara ECU sepeda motor dan sistem Android	ESP32 dengan koneksi wireless (Bluetooth/Wi-Fi) ke aplikasi Android	Latensi komunikasi wireless tinggi (400-600ms) yang berisiko menyebabkan kegagalan atau korupsi data saat proses flashing/remapping ECU.

No	Peneliti & Tahun	Fokus Penelitian	Metode	Analisis Celah (Gap)
10	(Firdaus & Artika, 2025)	Perancangan sistem identifikasi kelelahan berbasis perilaku berkendara melalui OBD-II	Pembacaan data PID standar via adapter ELM327	Hanya mampu membaca data PID standar OBD-II. Tidak memiliki fitur penulisan memori (write) dan tidak dapat mengakses parameter proprietary ECU Honda.
11	(Nguyen et al., 2025)	Mengembangkan perangkat diagnostik kendaraan jarak jauh menggunakan modul ESP32 yang berkomunikasi dengan sistem OBD-II kendaraan melalui protokol *CAN Bus*.	Aplikasi Android yang dikembangkan mampu melakukan pembacaan data sensor, pembacaan dan penghapusan DTC, serta pengujian aktivasi komponen dengan waktu respons 0,4-0,6 detik.	Penelitian ini menunjukkan tren pengembangan alat diagnostik berbasis *wireless*, namun waktu respons yang relatif tinggi menjadi kendala untuk operasi *flashing* yang memerlukan timing presisi.

Perkembangan teknologi diagnostik kendaraan berbasis *wireless* dan *Internet of Things* (IoT) telah menjadi tren dominan dalam dekade terakhir. (Nguyen et al., 2025) mengembangkan perangkat diagnostik kendaraan jarak jauh menggunakan modul ESP32 yang berkomunikasi dengan sistem *On-Board Diagnostics II* (OBD-II) kendaraan melalui protokol *Controller Area Network* (CAN Bus). Sistem yang dikembangkan mampu melakukan pembacaan data sensor secara *real-time*, pembacaan dan penghapusan *Diagnostic Trouble Code* (DTC), serta pengujian aktivasi komponen dengan waktu respons berkisar antara 0,4 hingga 0,6 detik melalui konektivitas *Firebase Realtime Database*. Meskipun penelitian ini menunjukkan keberhasilan dalam implementasi IoT untuk diagnostik otomotif, waktu respons yang relatif tinggi menjadi kendala signifikan apabila sistem digunakan untuk operasi *flashing* memori ECU yang mensyaratkan *timing* presisi tinggi sesuai standar ISO 14230-4.

Senada dengan pendekatan nirkabel, (Muslim & Subagio, 2025) merancang sistem antarmuka komunikasi data antara ECU sepeda motor Honda dengan aplikasi Android menggunakan mikrokontroler ESP32 sebagai *bridge* komunikasi. Sistem ini memanfaatkan protokol *K-Line* berbasis *Universal Asynchronous Receiver-Transmitter* (UART) untuk membaca data dari ECU dan mengirimkannya ke aplikasi Android melalui koneksi *Bluetooth*. Hasil pengujian pada Honda Vario 125 menunjukkan bahwa sistem mampu menampilkan parameter mesin seperti RPM, suhu, tegangan baterai, dan DTC dengan akurasi identik dengan *scanner* resmi Honda. Namun demikian, keterbatasan sistem terletak pada konektivitas *Bluetooth* yang memiliki latensi inheren berkisar 10 hingga 50 milidetik, serta jangkauan komunikasi yang terbatas pada radius kurang dari 10 meter, sehingga tidak ideal untuk proses *remapping* yang memerlukan transfer data dalam volume besar dengan keandalan tinggi.

Penelitian berbasis mikrokontroler untuk diagnostik sepeda motor juga dilakukan oleh beberapa peneliti dengan pendekatan yang bervariasi. (Mindarta et al., 2022) menerapkan teknologi *Diagnostic Tool* motor injeksi versi Android di Bengkel Sinar Mustika Motor menggunakan modul ELM327 yang terhubung ke *Data Link Connector* (DLC) motor. Sistem ini mampu membaca, mereset kode kesalahan, dan memanipulasi data konsumsi bahan bakar melalui aplikasi berbasis Android. Keberhasilan implementasi sistem tersebut membuktikan bahwa alat diagnostik berbasis *smartphone* dapat menjadi alternatif yang ekonomis bagi bengkel kecil. Meskipun demikian, penggunaan modul ELM327 yang beroperasi melalui protokol *Virtual COM Port* (VCP) memiliki keterbatasan dalam hal latensi komunikasi akibat *overhead* lapisan abstraksi *driver* Windows yang dapat mencapai 16 milidetik per transaksi.

(Sari & Wailanduw, 2022) mengembangkan alat *monitoring* kerja sensor pada sepeda motor injeksi berbasis Arduino Nano dengan kemampuan menyimpan data ke *spreadsheet* Excel melalui program pihak ketiga PLX-DAQ. Sistem ini dirancang untuk menggolah data dari sensor-sensor mesin dan menampilkannya pada *Liquid Crystal Display* (LCD) atau mengekspornya untuk analisis lebih lanjut. Pendekatan berbasis Arduino memberikan fleksibilitas dalam pengembangan dan biaya yang relatif rendah, namun keterbatasan kapabilitas *clock speed* mikrokontroler Arduino (16 MHz)

menyebabkan presisi *timing* yang tidak memadai untuk mengakomodasi persyaratan protokol *K-Line* ISO 9141-2 yang mensyaratkan *Fast Initialization* dengan toleransi waktu ±1 milidetik.

Penggunaan sensor eksternal untuk *monitoring* kendaraan juga masih banyak diterapkan dalam penelitian-penelitian terdahulu. (Sholeh et al., 2020) mengembangkan sistem *monitoring* kondisi kendaraan motor injeksi berbasis mikrokontroler yang memberikan peringatan kepada pemilik kendaraan melalui *Short Message Service* (SMS) *gateway*. Sistem ini memanfaatkan sensor kecepatan dan berbagai komponen elektronik untuk memantau waktu penggantian oli, air radiator, dan indikator kerusakan mesin. Pendekatan berbasis sensor eksternal memiliki keunggulan dalam hal kemudahan instalasi tanpa perlu mengakses sistem elektronik internal kendaraan, namun data yang diperoleh tidak berasal langsung dari ECU sehingga akurasi bergantung pada kualitas sensor dan rentan terhadap *noise* elektromagnetik dari sistem kelistrikan kendaraan.

Beberapa peneliti mengembangkan sistem monitoring menggunakan pendekatan sensor eksternal. (Septiandes et al., 2020) merancang alat pengukur RPM untuk sepeda motor injeksi menggunakan sensor induksi elektromagnetik yang mendeteksi pulsa dari sistem pengapian. Namun, pendekatan berbasis sensor eksternal ini memiliki keterbatasan karena data yang diperoleh tidak berasal langsung dari ECU dan rentan terhadap *noise* elektromagnetik dari sistem kelistrikan kendaraan.

Penelitian di bidang pengujian performa mesin pasca-modifikasi ECU juga menjadi fokus beberapa peneliti dari disiplin ilmu Teknik Mesin. (Witaszek, 2020) mengembangkan model prediksi konsumsi bahan bakar menggunakan *Artificial Neural Networks* (ANN) dengan data yang diperoleh dari antarmuka diagnostik OBD-II menggunakan adaptor ELM327. Penelitian ini menunjukkan bahwa data dari ECU dapat dimanfaatkan untuk analisis performa kendaraan, meskipun fokusnya lebih pada pemodelan matematis dibandingkan pengembangan sistem diagnostik itu sendiri. Di sisi lain, (Fauzi, 2021) merancang sistem penyalaan *engine* sepeda motor berbasis Arduino melalui koneksi *Bluetooth* Android dengan jangkauan maksimum 10 meter. Penelitian

tersebut membuktikan kelayakan komunikasi nirkabel untuk kendali kendaraan, namun tidak mengeksplorasi aspek diagnostik atau *remapping* ECU.

Pada ranah pemantauan perilaku pengemudi, (Firdaus & Artika, 2025) merancang sistem identifikasi kelelahan berbasis pembacaan data PID standar OBD-II menggunakan adapter ELM327. Meskipun sistem tersebut berhasil membaca parameter dasar kendaraan, keterbatasannya terletak pada ketidakmampuan mengakses parameter *proprietary* ECU Honda dan tidak memiliki fitur penulisan memori (*write*) yang diperlukan untuk proses *remapping*.

Dalam konteks sistem cerdas untuk diagnostik kerusakan, (Setyadi et al., 2024) membangun sistem berbasis *knowledge-based* untuk mendiagnosis kerusakan mesin motor roda tiga menggunakan metode *Case-Based Reasoning* dengan algoritma *K-Nearest Neighbor*. Sistem ini mampu memberikan rekomendasi perbaikan berdasarkan pola gejala kerusakan dengan tingkat akurasi 85% berdasarkan 20 sampel data diagnostik. Pendekatan *intelligent system* tersebut memberikan inspirasi untuk pengembangan fitur rekomendasi otomatis pada sistem diagnostik, meskipun belum terintegrasi dengan pembacaan data langsung dari ECU.

Penerapan kecerdasan buatan dalam domain otomotif tidak terbatas pada sistem diagnostik mesin. (Ristantyo et al., 2022) mengembangkan sistem identifikasi Tanda Nomor Kendaraan Bermotor (TNKB) berbasis *Convolutional Neural Network* dan *Optical Character Recognition* dengan akurasi 84,59%. Meskipun fokus penelitiannya berbeda, pendekatan *machine learning* yang digunakan membuka kemungkinan pengembangan fitur diagnostik cerdas yang dapat menganalisis pola data ECU secara otomatis pada penelitian lanjutan.

Penelitian terkait keamanan sistem ECU dan protokol komunikasi otomotif juga menjadi perhatian komunitas akademis internasional. (Holla & Akhila, 2020) mengimplementasikan layanan *Security Access* menggunakan algoritma SHA-2 untuk melindungi informasi dari sensor Radar pada kendaraan. Penelitian ini menekankan pentingnya mekanisme autentikasi *Seed-Key* dalam protokol *Unified Diagnostic Services* (UDS) untuk mencegah akses tidak terotorisasi ke sistem ECU. (Yassin et al., 2023)

mengusulkan sistem diagnostik otomotif jarak jauh berbasis teknologi *Blockchain* untuk menjamin keamanan dan integritas data kendaraan. Kedua penelitian tersebut menggarisbawahi aspek keamanan yang krusial dalam pengembangan sistem diagnostik modern, namun implementasinya memerlukan infrastruktur yang kompleks dan tidak selalu praktis untuk aplikasi bengkel skala kecil.

Berdasarkan analisis komprehensif terhadap sebelas penelitian terdahulu sebagaimana disajikan pada Tabel 2. 1, dapat diidentifikasi beberapa pola kelemahan umum yang menjadi celah penelitian. Pertama, sebagian besar penelitian masih mengandalkan koneksi nirkabel seperti *Bluetooth* atau *Wi-Fi* yang memiliki latensi inheren tidak deterministik, berkisar antara 10 hingga 600 milidetik bergantung pada kondisi *interference* dan jarak komunikasi. Latensi yang tinggi dan tidak dapat diprediksi ini sangat problematis untuk protokol *K-Line* sesuai standar ISO 9141-2 yang mensyaratkan *timing* inisialisasi dengan presisi tinggi, khususnya pada prosedur *Fast Initialization* yang memerlukan *break signal* selama tepat 25±1 milidetik.

Kedua, penelitian yang menggunakan koneksi kabel melalui metode *Virtual COM Port* juga menghadapi kendala latensi akibat *overhead* lapisan abstraksi *driver* sistem operasi Windows. Latensi *default* pada *driver* VCP dapat mencapai 16 milidetik per transaksi, ditambah dengan variabilitas yang disebabkan oleh mekanisme *scheduling* sistem operasi yang bersifat *preemptive*. Akumulasi latensi pada serangkaian transaksi komunikasi dapat menyebabkan kegagalan inisialisasi protokol atau, lebih kritis lagi, korupsi data saat proses penulisan memori ECU (*flashing*/*remapping*) yang berpotensi mengakibatkan kerusakan permanen (*bricking*) pada unit ECU.

Ketiga, mayoritas penelitian terdahulu hanya mengembangkan sistem dengan fungsi *read-only* yang terbatas pada pembacaan parameter sensor dan kode kesalahan tanpa kemampuan untuk melakukan modifikasi data kalibrasi ECU. Penelitian yang membahas *remapping* ECU umumnya menggunakan perangkat lunak komersial *proprietary* tanpa kendali terhadap algoritma komunikasi dan mekanisme validasi integritas data yang digunakan. Ketiadaan kontrol terhadap aspek-aspek tersebut

menyulitkan identifikasi dan penanganan potensi kesalahan selama proses transfer data ke memori *Flash* ECU.

Keempat, aspek keamanan data dalam proses *remapping* belum mendapat perhatian memadai pada penelitian-penelitian terdahulu. Mekanisme autentikasi *Seed-Key* yang merupakan persyaratan standar pada protokol KWP2000 untuk mengakses area memori *restricted* ECU tidak dibahas secara mendalam. Tanpa implementasi algoritma *Seed-Key* yang tepat, sistem tidak dapat melewati lapisan keamanan ECU dan tidak dapat melakukan operasi penulisan memori.

Dalam konteks pengembangan teknologi otomotif di lingkungan akademis Indonesia, penelitian terkait kendaraan bermotor telah dilakukan dari berbagai perspektif. Keberadaan penelitian-penelitian tersebut menunjukkan kapabilitas dan komitmen institusi pendidikan tinggi dalam pengembangan teknologi otomotif, sekaligus menegaskan perlunya pengembangan lebih lanjut khususnya dalam hal sistem diagnostik dan *remapping* ECU yang mandiri, ekonomis, dan dapat diandalkan untuk bengkel-bengkel independen.

Berdasarkan analisis celah penelitian tersebut, penelitian ini hadir untuk mengisi kekosongan dengan menawarkan solusi yang menggabungkan keunggulan dari berbagai pendekatan sebelumnya sembari mengatasi keterbatasan-keterbatasan yang teridentifikasi. Berbeda dengan penelitian-penelitian terdahulu yang menggunakan koneksi nirkabel atau *Virtual COM Port*, penelitian ini menerapkan metode *Direct Driver Access* melalui pustaka FTD2XX_NET.dll yang mengakses langsung *chipset* FTDI FT232RL tanpa melalui lapisan abstraksi *driver* Windows. Pendekatan ini memanfaatkan *Application Programming Interface* (API) tingkat rendah yang disediakan oleh FTDI untuk berkomunikasi secara langsung dengan *buffer* perangkat keras, sehingga mampu memangkas latensi komunikasi hingga kurang dari 2 milidetik per transaksi.

Keunggulan metode *Direct Driver Access* terletak pada kemampuannya untuk mengkonfigurasi parameter komunikasi dengan presisi tinggi, termasuk pengaturan *Latency Timer* hingga nilai minimum 1 milidetik dan kontrol langsung terhadap *bit-bang mode* untuk menghasilkan sinyal inisialisasi dengan *timing* yang deterministik. Dengan

latensi yang rendah dan dapat diprediksi, proses inisialisasi protokol *K-Line* sesuai standar ISO 9141-2 dapat dilakukan dengan tingkat keberhasilan yang mendekati 100%, jauh lebih reliabel dibandingkan metode VCP yang tingkat keberhasilannya seringkali di bawah 50% pada kondisi operasional nyata.

Selain aspek latensi, penelitian ini juga membangun aplikasi diagnostik dan *flasher* secara mandiri dengan kendali penuh terhadap seluruh aspek komunikasi dan keamanan data. Implementasi algoritma validasi *checksum* ganda, baik *checksum* 8-bit untuk paket komunikasi maupun *checksum* 32-bit untuk integritas keseluruhan file biner, menjamin bahwa data yang ditulis ke memori ECU adalah identik dengan file sumber tanpa mengalami korupsi selama proses transfer. Mekanisme autentikasi *Seed-Key* juga diimplementasikan untuk dapat melewati lapisan keamanan ECU dan mengakses area memori *restricted* yang diperlukan untuk operasi *remapping*.

Dengan demikian, kebaruan (*novelty*) penelitian ini terletak pada: pertama, penerapan metode *Direct Driver Access* untuk komunikasi *K-Line* dengan latensi rendah dan deterministik; kedua, pembangunan sistem diagnostik dan *flasher* ECU secara mandiri dengan kendali penuh terhadap protokol komunikasi; ketiga, implementasi mekanisme keamanan data berupa algoritma *checksum* dan autentikasi *Seed-Key* untuk proses *remapping* yang aman; serta keempat, validasi efektivitas sistem melalui pengujian *dynamometer* pada tiga tipe ECU berbeda, yaitu Keihin pada Honda BeAT FI CBS K25 dan Shindengen pada Honda Vario 125 serta Honda Vario 150.

2.2 Landasan Teori

Landasan teori dalam penelitian ini disusun sebagai kerangka acuan ilmiah untuk menganalisis permasalahan yang berkaitan dengan perancangan sistem diagnostik dan modifikasi memori (*remapping*) pada *Electronic Control Unit* (ECU) kendaraan roda dua. Teori-teori yang dikaji mencakup prinsip kerja sistem kendali elektronik, protokol komunikasi data serial, arsitektur *driver* perangkat keras, serta parameter termodinamika mesin. Rujukan yang digunakan bersumber dari literatur mutakhir dan standar internasional yang relevan dengan objek penelitian.

2.2.1 *Electronic Control Unit* (ECU) dan Sistem Injeksi Bahan Bakar Elektronik

2.2.1.1 Definisi dan Fungsi ECU

Electronic Control Unit (ECU) merupakan sistem kendali digital berbasis mikrokontroler yang mengatur seluruh aspek operasional mesin injeksi pada kendaraan bermotor. Pada sistem injeksi bahan bakar elektronik (*Electronic Fuel Injection*/EFI), ECU berfungsi sebagai otak yang menerima masukan dari berbagai sensor, memproses data berdasarkan algoritma kendali, dan menghasilkan sinyal keluaran untuk menggerakkan aktuator seperti injektor bahan bakar dan koil pengapian (Dibaei et al., 2020).

Arsitektur ECU sepeda motor injeksi umumnya mengadopsi mikrokontroler dengan prosesor Renesas SH-*series* atau sejenisnya yang beroperasi pada frekuensi 40-80 MHz. Sistem kendali bekerja berdasarkan siklus *input-process-output*: sinyal analog dari sensor dikonversi menjadi data digital melalui *Analog-to-Digital Converter* (ADC) internal, diolah melalui algoritma kendali berbasis *lookup table*, kemudian menghasilkan sinyal kendali terhadap aktuator (Fatra et al., 2023).

2.2.1.2 Arsitektur Memori Mikrokontroler

Mikrokontroler pada ECU menggunakan arsitektur memori terpisah dengan tiga area utama yang memiliki fungsi spesifik. Pemahaman terhadap organisasi memori ini menjadi prasyarat fundamental dalam melakukan modifikasi parameter ECU.

Flash Memory berukuran 48-512 KB berfungsi sebagai penyimpanan permanen untuk *firmware* dan data kalibrasi. Area ini menyimpan tabel *Fuel Map* dan *Ignition Map* yang menjadi basis algoritma kendali mesin. Proses *remapping* pada dasarnya merupakan operasi penghapusan (*erase*) dan penulisan ulang (*write*) area kalibrasi menggunakan protokol KWP2000 *Service* $34 (*Request Download*), $36 (*Transfer Data*), dan $37 (*Request Transfer Exit*) (Pratama et al., 2024).

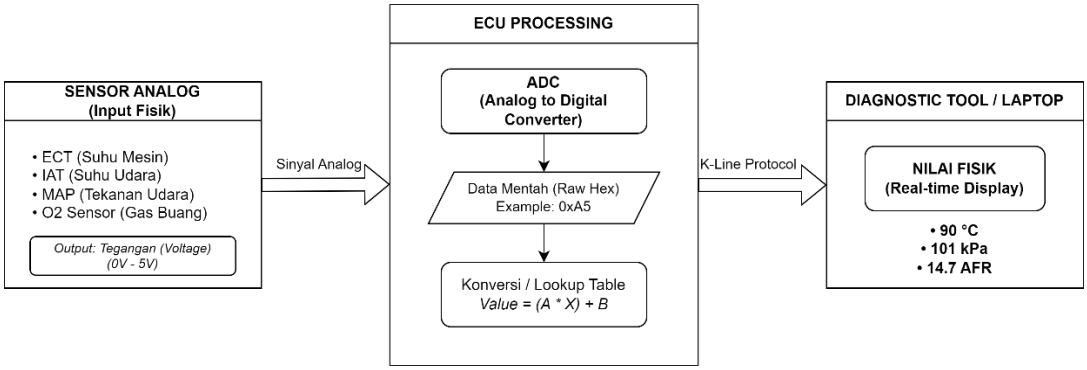
EEPROM (*Electrically Erasable Programmable Read-Only Memory*) berukuran 4-16 KB digunakan untuk menyimpan data yang berubah selama operasi mesin namun tidak boleh hilang saat daya terputus. Data yang tersimpan meliputi kode kesalahan

(*Diagnostic Trouble Code*/DTC), data adaptasi jangka panjang (*Long-Term Fuel Trim*/LTFT), dan konfigurasi *immobilizer*.

RAM (*Random Access Memory*) berukuran 16-32 KB berfungsi sebagai memori kerja tempat ECU menyimpan variabel sementara selama proses komputasi. RAM bersifat *volatile* dan seluruh isinya hilang saat ECU dimatikan, namun kecepatan aksesnya yang tinggi memungkinkan prosesor melakukan operasi baca-tulis dengan latensi minimal pada setiap siklus komputasi 10-20 ms.

2.2.1.3 Sensor Masukan dan Aktuator Keluaran

Sensor pada sistem injeksi bahan bakar elektronik menghasilkan sinyal analog yang mencerminkan kondisi fisis mesin. Sinyal tersebut dikonversi menjadi data digital melalui ADC 10-*bit* internal ECU dengan rentang nilai 0-1023 untuk tegangan masukan 0-5V. Diagram alur konversi sinyal sensor diilustrasikan pada Gambar 2. 1.



Gambar 2. 1 Diagram Alur Konversi Sinyal Sensor dari Analog ke Digital melalui ADC ECU

Throttle Position Sensor (TPS) menggunakan potensiometer tiga kabel yang menghasilkan tegangan proporsional terhadap sudut bukaan katup gas 0,5V pada kondisi *idle* penuh hingga 4,5V pada kondisi *wide-open throttle* (WOT).

Manifold Absolute Pressure (MAP) *Sensor* menghasilkan tegangan linear terhadap tekanan *intake manifold* (0,5V pada 20 kPa hingga 4,5V pada 100 kPa). Sensor ini digunakan bersama data RPM untuk menghitung massa udara masuk dalam sistem *speed-density*.

Engine Coolant Temperature (ECT) dan *Intake Air Temperature* (IAT) menggunakan termistor NTC (*Negative Temperature Coefficient*) yang resistansinya berbanding terbalik dengan suhu.

Oxygen Sensor (O_2) tipe zirkonia menghasilkan tegangan 0,1-0,9V berdasarkan konsentrasi oksigen di gas buang untuk koreksi AFR secara *closed-loop* (Gong et al., 2022).

Aktuator utama injektor bahan bakar dan koil pengapian dikontrol melalui sinyal *Pulse Width Modulation* (PWM) dan *trigger* digital berdasarkan hasil pencarian nilai di *lookup table* dengan masukan dari sensor-sensor tersebut.

2.2.2 Protokol Komunikasi ISO 9141-2 (*K-Line*)

2.2.2.1 Arsitektur Protokol K-Line

Komunikasi antara perangkat diagnostik eksternal dengan ECU kendaraan bermotor dilakukan melalui jalur serial tunggal yang dikenal sebagai K-Line. Protokol ini mengadopsi standar ISO 9141-2 untuk lapisan fisik (*physical layer*) dengan implementasi protokol aplikasi berbasis KWP2000 (*Keyword Protocol* 2000, ISO 14230-4). Arsitektur komunikasi menggunakan model *master-slave*, di mana perangkat eksternal bertindak sebagai *master* yang menginisiasi komunikasi, sedangkan ECU bertindak sebagai *slave* yang merespons perintah (Dibaei et al., 2020).

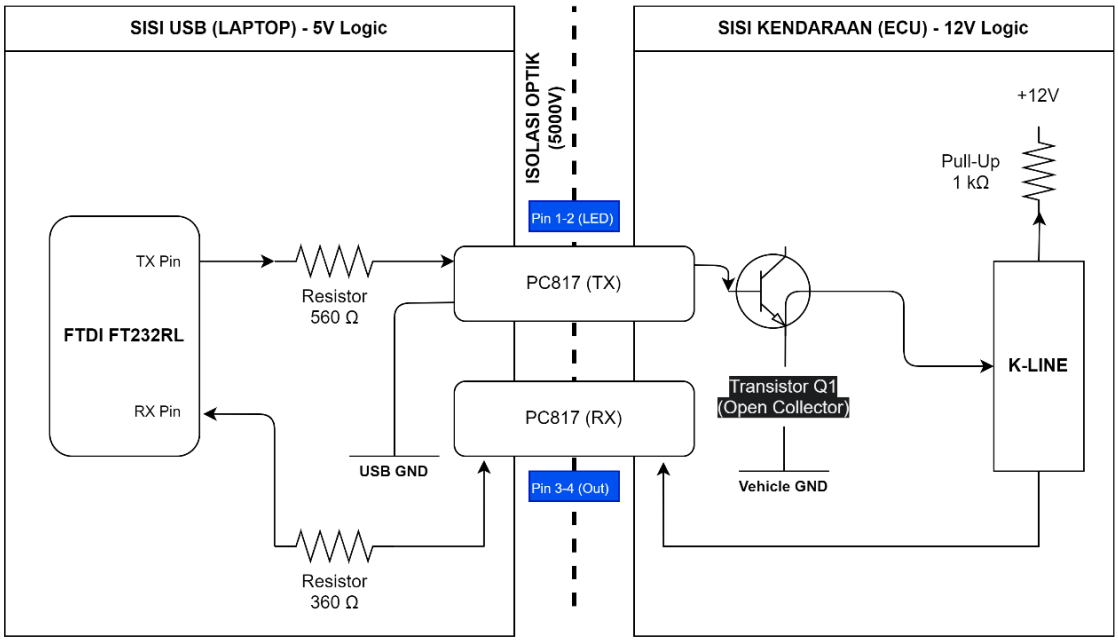
ISO 9141-2 menetapkan parameter komunikasi dasar: kecepatan transmisi 10.400 *baud*, format data 8-*bit* dengan 1 *stop bit* tanpa *parity*, serta metode inisialisasi yang memerlukan *timing* presisi tinggi. Karakteristik *half-duplex* pada K-Line berarti jalur yang sama digunakan secara bergantian untuk transmisi dan penerimaan data.

2.2.2.2 Lapisan Fisik *(Physical Layer)*

Standar ISO 9141-2 mendefinisikan level tegangan logika sebagai berikut: logika '1' (*high*) direpresentasikan oleh tegangan mendekati tegangan baterai kendaraan (umumnya 12V), sedangkan logika '0' (*low*) adalah tegangan mendekati *ground* (0V). Perbedaan level tegangan ini memerlukan rangkaian pengkondisi sinyal (*level shifter*)

untuk mengonversi sinyal TTL 5V dari antarmuka USB ke level tegangan 12V yang kompatibel dengan ECU (Kamath, 2022).

Rangkaian antarmuka fisik untuk komunikasi K-Line umumnya menggunakan *optocoupler* ganda untuk isolasi galvanik antara sisi USB (5V) dan sisi K-Line (12V). Isolasi galvanik berfungsi mencegah kerusakan akibat *ground loop* atau lonjakan tegangan dari sistem kelistrikan kendaraan (Editorial Staff, 2020). Skema rangkaian standar antarmuka USB-to-K-Line dengan isolasi *optocoupler* ditunjukkan pada Gambar 2. 2.

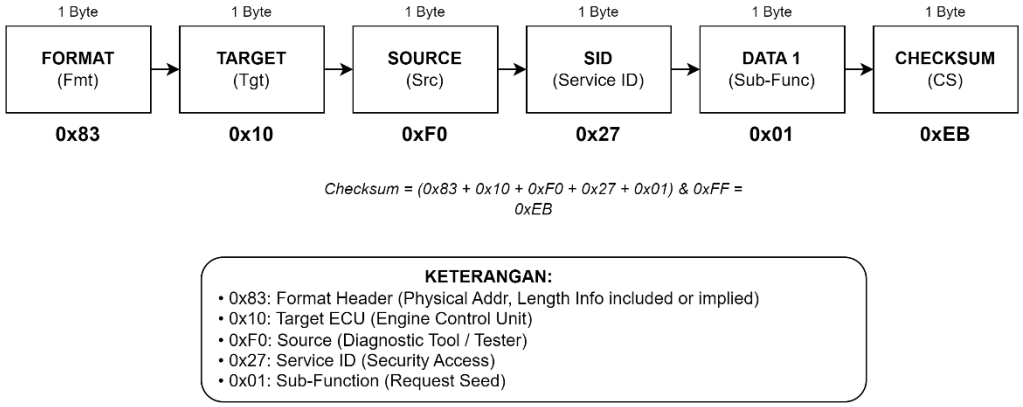


Gambar 2. 2 Skema Rangkaian Antarmuka Fisik USB-to-K-Line dengan Isolasi Optocoupler

Jalur transmisi (TX) dari antarmuka USB melewati resistor pembatas arus, LED indikator, dan *optocoupler* pertama sebelum terhubung ke jalur K-Line melalui transistor *open-collector*. Jalur penerimaan (RX) menggunakan *optocoupler* kedua dengan resistor *pull-up* pada sisi K-Line untuk memastikan jalur berada pada kondisi *idle* (logika '1' atau 12V) saat tidak ada transmisi.

2.2.2.3 Struktur Paket Data *(Frame Structure)*

Protokol KWP2000 menggunakan struktur paket data dengan format tertentu untuk memastikan integritas dan kemampuan deteksi kesalahan selama transmisi. Setiap *frame* data terdiri dari beberapa komponen utama yang diilustrasikan pada Gambar 2. 3.



Gambar 2. 3 Struktur Bingkai Data KWP2000 dengan Contoh Paket Security Access Request

Header berisi informasi format dan alamat. *Format Byte* mengkodekan panjang data (6 *bit*), *flag physical/functional addressing* (1 *bit*), dan *flag additional length* (1 *bit*).

Target Address dan *Source Address* mendefinisikan alamat tujuan dan pengirim. Untuk komunikasi dengan ECU, umumnya digunakan alamat 0x10 (ECU) sebagai target dan 0xF0 (*external tester*) sebagai *source*.

Service Identifier (SID) menentukan jenis operasi yang diminta, seperti 0x34 untuk *Request Download*, 0x36 untuk *Transfer Data*, dan 0x27 untuk *Security Access*.

Checksum dihitung dengan menjumlahkan semua *byte* sebelumnya dan mengambil komplemen 8-*bit*, memastikan bahwa penjumlahan seluruh *byte* dalam *frame* menghasilkan nilai 0x00 (Holla & Akhila, 2020).

2.2.3 Mekanisme Inisialisasi (*Fast Initialization*)

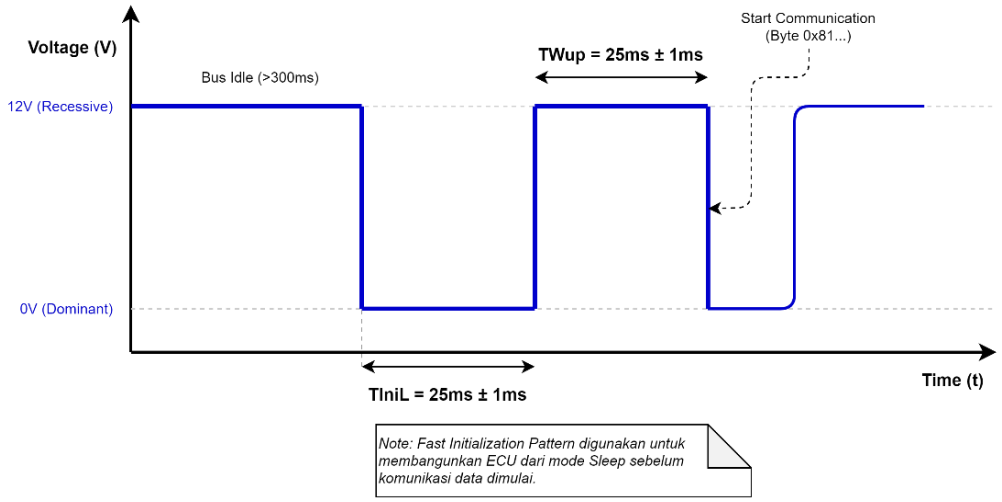
2.2.3.1 Prosedur Standar ISO 14230-4

Inisialisasi komunikasi antara perangkat diagnostik eksternal dan ECU merupakan prosedur kritis yang menentukan keberhasilan seluruh sesi diagnostik atau *flash programming*. Standar ISO 14230-4 mendefinisikan dua metode inisialisasi: *Slow Initialization* (5-*baud init*) yang menggunakan pertukaran *byte* pada kecepatan sangat rendah, dan *Fast Initialization* yang mengandalkan pola sinyal *timing* presisi tinggi tanpa memerlukan komunikasi lambat.

Fast Initialization dapat menyelesaikan proses inisialisasi dalam waktu kurang dari 100 ms jika dilakukan dengan benar, berbeda dengan *Slow Init* yang memerlukan waktu sekitar 5-10 detik untuk menyelesaikan *handshake*. Keberhasilan *Fast Initialization* sangat bergantung pada ketepatan *timing* sinyal yang dihasilkan oleh perangkat diagnostik dengan toleransi yang sangat ketat (±1 ms).

2.2.3.2 Diagram *Timing* dan Parameter Temporal

Prosedur *Fast Initialization* terdiri dari beberapa fase dengan durasi yang telah ditentukan secara spesifik oleh standar ISO 14230-4. Diagram waktu prosedur ini ditunjukkan pada Gambar 2. 4.



Gambar 2. 4 Diagram Waktu Fast Initialization dengan Durasi TIniL dan TWup

Fase Pertama: Kondisi Bus *Idle*

Jalur K-Line harus berada pada logika '1' (tegangan 12V) selama minimal 300 ms untuk memastikan ECU berada dalam kondisi siap dan tidak sedang memproses komunikasi sebelumnya.

Fase Kedua: Pola *Wake-up*

Jalur K-Line ditarik dari kondisi *idle* (logika '1', 12V) ke logika '0' (0V) dan dipertahankan selama durasi:

$$T_{IniL} = 25\ \text{ms} \pm 1\ \text{ms} \tag{2.2}$$

Setelah durasi T_{IniL} berakhir, jalur dikembalikan ke logika '1' (12V) dan dipertahankan selama durasi:

$$T_{Wup} = 25\ \text{ms} \pm 1\ \text{ms} \tag{2.3}$$

Kombinasi dua pulsa ini membentuk pola unik yang membedakan *Fast Init* dari gangguan acak atau *noise* pada jalur K-Line. ECU memiliki sirkuit deteksi *timing* internal yang hanya akan merespons jika kedua durasi TIniL dan TWup berada dalam toleransi yang ditentukan.

Fase Ketiga: Pesan *Start Communication*

Setelah pola *wake-up* selesai, perangkat diagnostik segera mengirim pesan *Start Communication* dengan format KWP2000. ECU akan merespons dengan pesan *Positive*

Response yang berisi *keybyte* 1 dan *keybyte* 2 yang mengindikasikan parameter komunikasi yang didukung.

2.2.3.3 Keterbatasan *Virtual COM Port* (VCP)

Virtual COM Port (VCP) merupakan mode operasi *default* pada *driver* FTDI yang menyediakan antarmuka emulasi *port* serial standar untuk aplikasi Windows. Mode ini memungkinkan aplikasi menggunakan fungsi Win32 API standar untuk berkomunikasi dengan perangkat USB-to-Serial. Namun, arsitektur VCP memiliki beberapa keterbatasan fundamental yang membuatnya tidak cocok untuk implementasi *Fast Initialization* yang memerlukan presisi *timing* tinggi (Fauzi, 2021).

Keterbatasan Pertama: *Latency Timer*

Driver FTDI menggunakan nilai *latency timer default* 16 ms, yang berarti data yang diterima dari USB akan ditahan (*buffered*) hingga 16 ms sebelum diteruskan ke aplikasi. Untuk aplikasi yang memerlukan respons *real-time* seperti *Fast Init*, latensi ini menyebabkan keterlambatan antara saat data dikirim oleh aplikasi dan saat data benar-benar muncul pada jalur TX fisik (FTDI Chip, 2023).

Keterbatasan Kedua: Presisi *Timing* Fungsi *Sleep*

Fungsi Thread.Sleep() pada C# atau Sleep() pada Win32 API memiliki resolusi yang ditentukan oleh *scheduler quantum* Windows, yang umumnya berkisar 10-15 ms pada sistem operasi versi *desktop*. Variasi ini jauh melebihi toleransi ±1 ms yang dipersyaratkan oleh ISO 14230-4.

Keterbatasan Ketiga: Kontrol Sinyal RTS/DTR

Perubahan status sinyal kontrol RTS (*Request to Send*) dan DTR (*Data Terminal Ready*) pada mode VCP harus melewati beberapa lapisan abstraksi: aplikasi → *driver* VCP → *driver* USB → *hardware* FTDI → pin fisik. Setiap lapisan menambahkan latensi dan ketidakpastian *timing*.

2.2.4 *Direct Driver Access* dan Pustaka FTDI D2XX

2.2.4.1 Konsep *Direct Driver Access*

Untuk mengatasi keterbatasan VCP, pendekatan *Direct Driver Access* melalui FTDI D2XX API dapat digunakan. D2XX API menyediakan akses tingkat rendah (*low-level access*) langsung ke perangkat FTDI tanpa melalui lapisan abstraksi VCP, memungkinkan kontrol penuh terhadap *buffer* transmisi/penerimaan, *timing* transfer, dan mode operasi *hardware* (FTDI Chip, 2023).

Fungsi-fungsi D2XX seperti FT_Open(), FT_Write(), FT_Read(), dan FT_SetBitMode() memanipulasi perangkat FTDI secara langsung melalui *bulk transfer* USB, melewati *buffering* sistem operasi dan mencapai latensi yang jauh lebih rendah (<2 ms) dibandingkan VCP.

2.2.4.2 Mekanisme *Wrapper Driver* dan *Managed Code*

Dalam arsitektur pengembangan aplikasi *desktop* berbasis .NET Framework (C#), komunikasi dengan perangkat keras memerlukan mekanisme penjembatan antara *managed code* yang dieksekusi oleh *Common Language Runtime* (CLR) dengan *unmanaged code* berupa pustaka biner (*native library*) yang mengakses register perangkat keras secara langsung.

Mekanisme *Platform Invocation Services* (P/Invoke) merupakan fitur .NET Framework untuk memanggil fungsi-fungsi dari *Dynamic Link Library* (DLL) yang dikompilasi dalam kode *native* (Win32 API). Setiap metode dalam pustaka *wrapper* memetakan parameter *managed* (.NET *types* seperti *string*, *byte*[], *int*) ke format parameter *unmanaged* (*pointer*, struktur memori *native*), melakukan pemanggilan fungsi D2XX, dan mengonversi nilai balik dari format *native* kembali ke tipe data *managed*.

2.2.4.3 Konfigurasi Parameter Kritis

Keunggulan pendekatan *Direct Driver Access* terletak pada kemampuan mengonfigurasi parameter krusial yang tidak dapat diakses melalui abstraksi VCP.

Latency Timer dapat dikonfigurasi hingga minimum 1 ms menggunakan fungsi FT_SetLatencyTimer(), sehingga memenuhi persyaratan waktu respons protokol KWP2000 yang mensyaratkan waktu respons P2 dalam rentang 25-50 ms.

Baud Rate Non-Standar seperti 10.400 *bps* sesuai persyaratan ISO 9141-2 dapat dikonfigurasi melalui fungsi FT_SetBaudRate(). Nilai pembagi (*divisor*) dihitung berdasarkan frekuensi *clock* internal FTDI menggunakan persamaan:

$$\text{Divisor} = \frac{f_{clk}}{16 \times \text{Baud Rate}} \tag{2.4}$$

dengan:

1. f_{clk} = frekuensi *clock* internal perangkat FTDI (3.000.000 Hz)
2. Baud Rate = kecepatan komunikasi yang diinginkan (10.400 bps)
3. Divisor = nilai pembagi bilangan bulat yang diprogram ke *register baud rate*

Bit-Bang Mode memungkinkan pin I/O FTDI beroperasi sebagai GPIO (*General Purpose Input/Output*) yang dapat dikontrol secara manual. Mode ini esensial untuk menghasilkan pola sinyal inisialisasi dengan *timing* presisi tinggi yang tidak dapat dicapai melalui mode UART standar.

2.2.4.4 Perbandingan VCP dan *Direct Driver Access*

Perbandingan karakteristik kedua metode akses disajikan pada Tabel 2. 2.

Tabel 2. 2 Perbandingan Metode Inisialisasi *K-Line*

Parameter	Virtual COM Port (VCP)	Direct Driver Access (D2XX)	Standar ISO 14230-4
Resolusi *Timer*	10 - 15 ms	< 1 ms	-
Latensi *Driver*	16 ms (*default*)	< 2 ms	-
Presisi T_{IniL}	20 - 35 ms	25.0 ± 0.3 ms	25 ± 1 ms

Parameter	Virtual COM Port (VCP)	Direct Driver Access (D2XX)	Standar ISO 14230-4
Tingkat Keberhasilan	< 50%	> 95%	-

2.2.5 Konsep *Remapping* dan Validasi Integritas Data

2.2.5.1 Definisi dan Prinsip *Remapping* ECU

Remapping atau *ECU tuning* merupakan proses modifikasi nilai-nilai numerik dalam tabel kalibrasi (*lookup table*) yang tersimpan di *flash memory* ECU untuk mengubah perilaku mesin sesuai tujuan kalibrasi tertentu. Modifikasi dilakukan menggunakan perangkat lunak editor ECU dengan file definisi berformat XDF (*eXtensible Definition File*) yang menerjemahkan kode heksadesimal mentah menjadi tabel-tabel yang dapat dibaca (Fatra et al., 2023).

Kerentanan pada mekanisme autentikasi *Seed-Key* menjadi perhatian serius dalam keamanan sistem diagnostik otomotif. (National Motor Freight Traffic Association, 2025) dalam publikasi teknisnya mengungkap bahwa penyerang dapat mengeksploitasi kelemahan protokol *Seed-Key* melalui serangan *reset* yang memaksa ECU menghasilkan *seed* yang dapat diprediksi, sehingga mereduksi mekanisme keamanan menjadi serangan *replay* sederhana. Temuan ini menggarisbawahi pentingnya implementasi algoritma *Seed-Key* yang robust pada sistem *flasher* ECU.

Tabel kalibrasi utama yang menjadi target modifikasi meliputi:

Fuel Map (*Peta Bahan Bakar*): Tabel multidimensi yang mendefinisikan durasi semprotan injektor (*injector pulse width*) sebagai fungsi dari putaran mesin (RPM) dan beban mesin. Target AFR bervariasi: 14,7:1 pada kondisi *cruise* untuk efisiensi, 13,0-13,5:1 pada WOT untuk tenaga maksimal, atau 12,5:1 pada *cold start* untuk stabilitas pembakaran.

Ignition Timing Map (*Peta Pengapian*): Tabel yang mengatur waktu pemercikan busi relatif terhadap posisi piston, dinyatakan dalam satuan derajat sebelum Titik Mati

Atas (°BTDC). Optimasi *ignition timing* bertujuan mencapai kondisi *Maximum Brake Torque* (MBT) *timing*.

Rev Limiter (*Pembatas Putaran*): Konstanta numerik yang menentukan ambang batas putaran maksimum mesin untuk mencegah kerusakan mekanis akibat gaya inersia berlebihan atau kegagalan *valve float*.

2.2.5.2 Algoritma Interpolasi *Lookup Table*

Ketika mesin beroperasi pada titik operasi tertentu yang tidak tepat berada pada titik *grid* yang terdefinisi dalam tabel, ECU menghitung nilai keluaran melalui algoritma interpolasi bilinear:

$$V(x,y) = \frac{1}{(x_2 - x_1)(y_2 - y_1)} [x_2 - x \quad x - x_1] \begin{bmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{bmatrix} \begin{bmatrix} y_2 - y \\ y - y_1 \end{bmatrix} \qquad (2.10)$$

dengan:

1. $V(x,y)$ = nilai interpolasi pada titik operasi (x,y)
2. x = RPM aktual mesin
3. y = posisi *throttle* aktual
4. x_1, x_2 = titik *grid* RPM yang mengapit x ($x_1 < x < x_2$)
5. y_1, y_2 = titik *grid throttle* yang mengapit y ($y_1 < y < y_2$)

2.2.5.3 Algoritma Validasi Integritas Data *(Checksum)*

Integritas data merupakan properti fundamental yang menjamin bahwa data tidak mengalami perubahan yang tidak terotorisasi akibat kesalahan perangkat keras, interferensi elektromagnetik, atau kegagalan komunikasi. Dalam konteks *remapping* ECU, validasi integritas data menjadi kritis karena kesalahan sekecil satu *bit* dalam file biner dapat menyebabkan kegagalan fungsi ECU secara permanen (*bricking*) (Pratama et al., 2024).

Checksum merupakan metode matematis untuk mendeteksi kesalahan data dengan menghitung nilai numerik tunggal yang merepresentasikan seluruh isi data. Persamaan matematis untuk *checksum* 8-*bit* sederhana dinyatakan sebagai:

$$C = \left(\sum_{i=0}^{n-1} D_i \right) \text{mod} 256 \qquad (2.5)$$

dengan:

1. C = nilai *checksum* (0-255)
2. D_i = nilai *byte* ke-i dalam blok data (0-255)
3. n = jumlah total *byte* dalam blok data
4. mod = operasi modulo (sisa pembagian)

Varian *checksum* yang lebih *robust* adalah *two's complement checksum*, di mana nilai *checksum* yang disimpan adalah komplemen dua dari hasil penjumlahan, sehingga penjumlahan seluruh data termasuk *byte checksum* akan menghasilkan nilai nol:

$$C = (256 - S) \text{mod} 256$$

dengan:

$$S = \left(\sum_{i=0}^{n-1} D_i \right) \text{mod} 256$$

Aspek keamanan pada sistem diagnostik kendaraan menjadi perhatian penting dalam pengembangan alat diagnostik dan *flasher* ECU. (Daily & Kulkarni, 2020) dalam penelitiannya tentang keamanan diagnostik kendaraan berat mengidentifikasi beberapa *attack vectors* yang dapat mengancam integritas sistem, meliputi perangkat lunak berbahaya pada PC diagnostik, manipulasi *driver* DLL, dan modifikasi *firmware* pada *Vehicle Diagnostic Adapter*. Penelitian tersebut merekomendasikan implementasi *Secure Gateway* sebagai lapisan perlindungan tambahan antara port diagnostik dengan jaringan internal ECU kendaraan.

Protokol KWP2000 mengimplementasikan validasi *checksum* pada tingkat pesan (*message-level checksum*). ECU akan menolak pesan dengan *checksum* yang tidak valid dengan mengirimkan *Negative Response Code* (NRC) yang mengindikasikan jenis kesalahan. Daftar kode kesalahan standar protokol KWP2000 disajikan pada Tabel 2. 3.

Tabel 2. 3 Kode Kesalahan *Checksum* pada Protokol KWP2000

Kode (Hex)	Nama Negative Response	Deskripsi Kesalahan	Severity	Tindakan
0x7F	Negative Response	Header untuk semua respons kesalahan dari ECU	Info	—
0x10	generalReject	ECU menolak request karena alasan umum (termasuk checksum error)	Critical	Retry
0x13	incorrectMessageLength	Panjang pesan tidak sesuai dengan yang diharapkan ECU	High	Validasi
0x21	busyRepeatRequest	ECU sedang sibuk, ulangi request setelah delay	Medium	Retry
0x22	conditionsNotCorrect	Kondisi prasyarat tidak terpenuhi untuk eksekusi service	Medium	Cek State
0x31	requestOutOfRange	Parameter atau alamat memori di luar range yang valid	High	Validasi

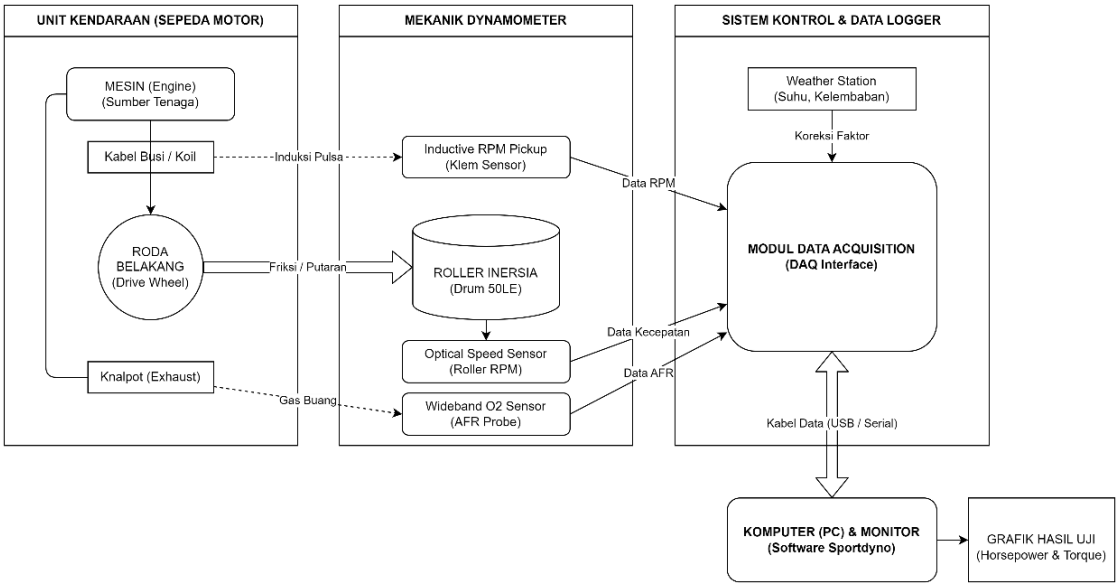
Kode (Hex)	Nama Negative Response	Deskripsi Kesalahan	Severity	Tindakan
0x33	securityAccessDenied	Akses ditolak karena belum authenticated (seed-key gagal)	Critical	Re-auth
0x35	invalidKey	Key yang dikirim tidak cocok dengan seed (security access gagal)	Critical	Re-calc
0x72	generalProgrammingFailure	Kegagalan umum flash programming (checksum ROM error)	Critical	Re-flash
0x78	requestCorrectlyReceived	Request diterima tapi butuh waktu proses (bukan error)	Info	Wait

2.2.6 Parameter Unjuk Kerja Mesin dan Metodologi Pengujian

2.2.6.1 Prinsip Kerja *Chassis Dynamometer*

Chassis dynamometer merupakan instrumen pengujian yang mengukur unjuk kerja mesin kendaraan bermotor pada kondisi stasioner dengan mensimulasikan beban jalan sebenarnya melalui mekanisme transfer daya dari roda penggerak ke *drum roller*. Prinsip pengukuran didasarkan pada hukum Newton kedua, di mana gaya yang diberikan roda penggerak terhadap *drum* dengan momen inersia tertentu menghasilkan percepatan sudut yang dapat dikonversi menjadi nilai torsi dan daya (Damayanti & Septiyanto, 2024).

Skema alat uji *chassis dynamometer* diilustrasikan pada Gambar 2. 5.



Gambar 2. 5 Skema Alat Uji Chassis Dynamometer Super Dyno 50LE

Inertia dynamometer merekam kurva akselerasi *drum roller* dengan momen inersia tertentu selama pengujian akselerasi penuh dari putaran rendah hingga *rev limiter* mesin. Sensor *optical encoder* atau *magnetic pickup* pada poros *drum* mendeteksi kecepatan angular dengan resolusi tinggi, yang kemudian diolah oleh perangkat lunak akuisisi data untuk menghitung torsi berdasarkan laju perubahan kecepatan angular.

Persamaan dasar pengukuran torsi pada *inertia dynamometer* dinyatakan sebagai:

$$T = I \times \alpha \tag{2.6}$$

dengan:

1. T = torsi pada roda penggerak (Nm)
2. I = momen inersia total *drum roller* $\text{kg} \cdot \text{m}^2$
3. α = percepatan angular *drum* $\frac{\text{rad}}{\text{s}^2}$

2.2.6.2 Hubungan Torsi dan Daya

Torsi merupakan besaran vektor yang menggambarkan gaya putar atau momen gaya yang dihasilkan oleh mekanisme engkol-piston sebagai hasil dari tekanan pembakaran campuran udara-bahan bakar di dalam ruang bakar. Secara fisis, torsi berbanding lurus dengan efisiensi volumetrik mesin, yaitu kemampuan mesin untuk menghisap massa udara maksimal ke dalam silinder pada setiap siklus hisap (Pratama et al., 2024).

Daya mesin didefinisikan sebagai laju usaha yang dilakukan oleh mesin dalam satuan waktu. Daya merupakan fungsi dari torsi dan kecepatan putar mesin, sehingga mesin dapat menghasilkan daya tinggi baik melalui torsi besar pada putaran rendah, maupun torsi sedang pada putaran sangat tinggi. Hubungan matematis antara torsi dan daya dalam satuan imperial dinyatakan:

$$\text{BHP} = \frac{\text{Torque} \times \text{RPM}}{5252} \tag{2.1}$$

dengan:

1. BHP = Brake Horsepower (daya mesin)
2. *Torque* = torsi mesin (lb-ft)
3. RPM = putaran mesin per menit
4. 5252 = konstanta konversi yang berasal dari $\frac{33.000}{(2\pi)}$

Untuk satuan metrik (daya dalam kW, torsi dalam Nm), persamaan setara yang digunakan adalah:

$$P_{kW} = \frac{T_{Nm} \times RPM}{9549} \quad (2.8)$$

dengan:

1. P_{kW} = daya dalam satuan Kilowatt (kW)
2. T_{Nm} = torsi dalam satuan Newton-meter (Nm)
3. 9549 = konstanta konversi untuk satuan metrik

2.2.6.3 Rasio Udara-Bahan Bakar (*Air-Fuel Ratio*/AFR)

Air-Fuel Ratio (AFR) merupakan perbandingan massa udara terhadap massa bahan bakar dalam campuran yang masuk ke ruang bakar. Nilai AFR stoikiometrik untuk bensin adalah 14,7:1, yang berarti diperlukan 14,7 gram udara untuk membakar sempurna 1 gram bensin. Kondisi AFR mempengaruhi karakteristik pembakaran:

1. AFR < 14,7:1 (*rich mixture*): Campuran kaya bahan bakar, menghasilkan tenaga lebih tinggi namun efisiensi bahan bakar lebih rendah. Digunakan pada kondisi beban tinggi dan WOT.
2. AFR = 14,7:1 (*stoichiometric*): Campuran ideal untuk pembakaran sempurna dengan emisi minimal. Digunakan pada kondisi *cruise* normal.
3. AFR > 14,7:1 (*lean mixture*): Campuran miskin bahan bakar, meningkatkan efisiensi namun dapat menyebabkan suhu pembakaran berlebih. Penggunaan harus hati-hati untuk menghindari kerusakan mesin.

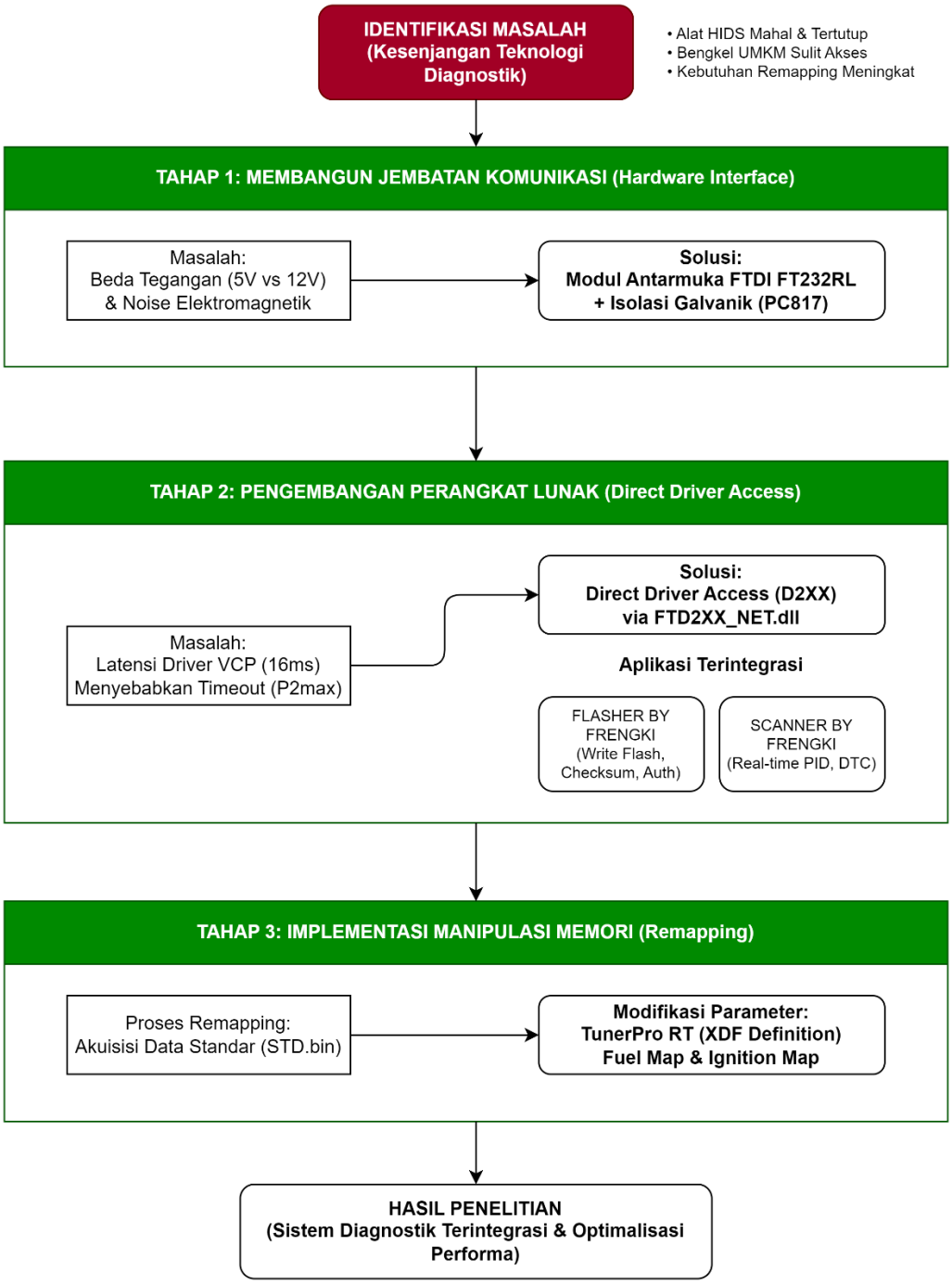
Sensor O_2 (*oxygen sensor*) pada sistem *exhaust* memberikan umpan balik kepada ECU mengenai kondisi AFR aktual. Pada mode operasi *closed-loop*, ECU melakukan koreksi durasi injeksi secara *real-time* berdasarkan pembacaan sensor O_2 untuk mempertahankan AFR mendekati nilai target yang ditentukan dalam *fuel map* (Gong et al., 2022).

2.3 Kerangka Pemikiran

Kerangka pemikiran dalam penelitian ini dibangun berdasarkan analisis kesenjangan (*gap analysis*) antara kebutuhan industri bengkel sepeda motor dengan ketersediaan alat diagnostik yang terjangkau. Sebagaimana telah diuraikan pada latar belakang, bengkel independen dan pelaku usaha kecil menengah (UMKM) di Indonesia menghadapi kendala signifikan dalam mengakses peralatan diagnostik resmi Honda, yang dikenal sebagai *Honda Intelligent Diagnostic System* (HIDS). Alat tersebut memiliki sifat tertutup (*proprietary*) dan harga yang relatif mahal, sehingga membatasi kemampuan bengkel non-resmi dalam melakukan diagnosis mendalam dan optimasi performa mesin melalui proses *remapping* ECU.

Di sisi lain, terdapat fakta teknis yang menjadi titik tolak penelitian ini, yaitu bahwa komunikasi antara alat diagnostik dengan *Electronic Control Unit* (ECU) sepeda motor Honda sesungguhnya dibangun di atas protokol standar terbuka yang telah didokumentasikan secara internasional. Standar ISO 9141-2 mendefinisikan persyaratan untuk pertukaran informasi digital antara ECU *on-board* dengan alat pemindai diagnostik, termasuk spesifikasi lapisan fisik K-Line dan prosedur inisialisasi komunikasi. Protokol ini kemudian dikembangkan lebih lanjut dalam standar ISO 14230-4:2000 atau *Keyword Protocol 2000* (KWP 2000) yang menetapkan format *header*, *timing parameter*, dan layanan diagnostik yang kompatibel dengan ECU modern Honda. Keterbukaan standar ini membuka peluang bagi pengembangan alat diagnostik alternatif yang mampu berkomunikasi dengan ECU Honda tanpa melanggar hak kekayaan intelektual pabrikan.

Berdasarkan premis tersebut, penelitian ini mengembangkan kerangka pemikiran yang sistematis untuk membangun sistem diagnostik dan *remapping* ECU yang terintegrasi. Alur logika penyelesaian masalah dalam penelitian ini terbagi menjadi tiga tahapan utama yang saling berkaitan, sebagaimana divisualisasikan pada Gambar 2. 6.



Gambar 2. 6 Bagan Alir Kerangka Pemikiran Penelitian

Tahap Pertama: Membangun Jembatan Komunikasi (Hardware Interface)

Tahap awal dalam kerangka pemikiran ini adalah membangun jembatan komunikasi fisik antara komputer dengan ECU sepeda motor. Tantangan utama pada tahap ini adalah perbedaan level tegangan operasional: komputer berbasis USB

beroperasi pada tegangan logika 5V, sedangkan sistem kelistrikan sepeda motor menggunakan tegangan 12V–14V. Selain itu, lingkungan kelistrikan kendaraan bermotor mengandung *noise* elektromagnetik yang tinggi, terutama dari sistem pengapian (*ignition coil*) yang dapat menghasilkan lonjakan tegangan hingga puluhan kilovolt.

Solusi yang diusulkan adalah perancangan modul antarmuka (*interface module*) berbasis *chipset* FTDI FT232RL yang dilengkapi dengan sirkuit isolasi galvanik menggunakan *optocoupler* PC817. Isolasi galvanik ini berfungsi sebagai pemisah total antara domain tegangan rendah (USB) dengan domain tegangan tinggi (K-Line motor), sehingga mencegah arus balik (*back EMF*) yang berpotensi merusak port USB komputer. Desain fisik modul ini dapat dilihat pada skema modul.jpeg yang menjadi acuan implementasi perangkat keras dalam penelitian ini.

Tahap Kedua: Pengembangan Perangkat Lunak dengan Akses Langsung (Direct Driver Access)

Tahap kedua berfokus pada pengembangan perangkat lunak yang mampu mengatasi keterbatasan latensi driver konvensional. Sebagaimana telah dibahas dalam tinjauan pustaka, penggunaan *Virtual COM Port* (VCP) standar Windows menghasilkan latensi rata-rata 16 milidetik per siklus data, yang seringkali menyebabkan *timeout* pada proses *flash programming* karena melebihi batas waktu respons (P2max) yang ditetapkan protokol ISO 14230-4 (FTDI Chip, 2023). Penelitian ini mengadopsi pendekatan *Direct Driver Access* (D2XX) yang memotong jalur *scheduler* sistem operasi dan mengakses langsung *buffer* USB melalui pustaka *FTD2XX_NET.dll* sebuah *driver wrapper* untuk platform .NET Framework yang dikembangkan oleh FTDI.

Implementasi perangkat lunak dalam penelitian ini terbagi menjadi dua aplikasi yang saling melengkapi:

1. FLASHER BY FRENGKI

   Aplikasi yang berfungsi untuk menulis data kalibrasi baru ke memori *flash* ECU. Aplikasi ini menerima *input* berupa file binari yang telah dimodifikasi (misalnya *putu.bin* atau *30400-K60R-B61_skripsi.bin*) dan menuliskannya ke

alamat memori ECU yang sesuai melalui protokol *Request Download* dan *Transfer Data*. Fitur *seed-key authentication* diterapkan untuk membuka akses memori yang dilindungi, sementara validasi *checksum Two's Complement* memastikan integritas setiap blok data yang ditransfer.

2. SCANNER BY FRENGKI

   Aplikasi yang berfungsi untuk memantau parameter sensor ECU secara *real-time* dan membaca riwayat kode kesalahan (*Diagnostic Trouble Code*/DTC). Aplikasi ini mengirimkan permintaan *Parameter Identification* (PID) ke ECU dan menampilkan nilai sensor seperti putaran mesin (RPM), posisi katup gas (TPS), suhu mesin (ECT), suhu udara masuk (IAT), dan tegangan baterai dalam antarmuka yang mudah dipahami oleh mekanik.

Tahap Ketiga: Implementasi Manipulasi Memori (Remapping)

Tahap ketiga merupakan implementasi fungsional dari sistem yang telah dibangun, yaitu proses *remapping* atau modifikasi parameter kalibrasi ECU untuk mengoptimalkan performa mesin. Proses ini dimulai dengan akuisisi data kalibrasi standar pabrik yang tersimpan dalam file binari (misalnya *STD.bin* untuk konfigurasi standar Honda BeAT FI K25, *30400-K60R-B61_std.bin* untuk Vario 125 K60R, dan *VARIO-150-K59F-A01 ORI.bin* untuk Vario 150 K59). File binari tersebut kemudian dibuka dan dimodifikasi menggunakan perangkat lunak editor ECU pihak ketiga, yaitu TunerPro RT, yang mampu menerjemahkan data heksadesimal mentah menjadi tabel-tabel peta bahan bakar (*Fuel Map*) dan peta pengapian (*Ignition Map*) yang dapat diedit secara visual.

Modifikasi yang dilakukan meliputi penyesuaian durasi injeksi bahan bakar pada berbagai kondisi RPM dan beban mesin untuk mencapai rasio campuran udara-bahan bakar (*Air-Fuel Ratio*/AFR) yang optimal, penyesuaian sudut pengapian (*ignition timing advance*) untuk memaksimalkan efisiensi pembakaran, serta perubahan batas putaran mesin (*RPM limiter*) untuk memungkinkan mesin beroperasi pada rentang putaran yang lebih tinggi. Hasil modifikasi kemudian disimpan sebagai file binari baru (misalnya *putu.bin* atau *skripsi.bin*) yang siap ditulis ke ECU menggunakan aplikasi *FLASHER BY FRENGKI*.

Sebagaimana terlihat pada Gambar 2. 6, proses dalam kerangka pemikiran ini mengikuti alur yang sistematis dimulai dari identifikasi masalah (keterbatasan alat diagnostik konvensional), dilanjutkan dengan akuisisi data ECU standar, modifikasi parameter kalibrasi menggunakan TunerPro, penulisan data baru ke ECU melalui aplikasi *Flasher* yang dikembangkan, verifikasi fungsi melalui pembacaan data *real-time* pada aplikasi *Scanner*, dan diakhiri dengan validasi performa mesin menggunakan alat uji dinamometer.

Validasi Keberhasilan Kerangka Pemikiran

Keberhasilan kerangka pemikiran dalam penelitian ini dibuktikan melalui dua indikator utama yang saling melengkapi:

1. Validasi Fungsional (Functional Validation)
   Indikator pertama adalah terbukanya akses terhadap parameter diagnostik ECU melalui aplikasi *SCANNER BY FRENGKI*. Keberhasilan ditunjukkan oleh kemampuan sistem untuk menampilkan data sensor secara *real-time* (RPM, TPS, ECT, IAT, tegangan baterai) dengan stabilitas sinyal yang baik tanpa *lag* atau *jitter* yang signifikan serta kemampuan membaca dan menghapus kode kesalahan (DTC) yang tersimpan dalam memori ECU.
2. Validasi Performa (Performance Validation)
   Indikator kedua adalah peningkatan performa mekanis mesin yang terukur secara kuantitatif melalui pengujian dinamometer (*dynotest*). Validasi dilakukan menggunakan alat uji *chassis dynamometer* Super Dyno 50LE untuk mengukur daya (*horsepower*) dan torsi (*torque*) mesin sebelum dan sesudah proses *remapping*. Data empiris menunjukkan peningkatan daya sebesar kurang lebih 1,58 HP (dari 7,25 HP menjadi 8,83 HP) pada sepeda motor Honda BeAT FI K25, yang mengonfirmasi efektivitas modifikasi parameter ECU yang dilakukan.

Dengan demikian, kerangka pemikiran yang dibangun dalam penelitian ini bertujuan untuk membuktikan hipotesis bahwa sistem diagnostik dan *remapping* ECU yang dikembangkan secara mandiri dengan biaya terjangkau mampu memberikan hasil

yang setara dengan peralatan komersial berbiaya tinggi. Kesetaraan tersebut diukur dari dua aspek: (1) akurasi pembacaan data diagnostik yang tercermin dari stabilitas tampilan parameter sensor, dan (2) efektivitas modifikasi ECU yang tercermin dari peningkatan performa mesin yang terukur pada alat uji dinamometer. Apabila kedua indikator tersebut terpenuhi, maka dapat disimpulkan bahwa kombinasi antara perancangan perangkat keras berbasis isolasi galvanik, pengembangan perangkat lunak berbasis *Direct Driver Access* (D2XX), dan metodologi *remapping* yang sistematis merupakan solusi yang viable bagi bengkel independen dan UMKM di Indonesia untuk mengakses teknologi optimasi ECU sepeda motor Honda.