



**AGH**

**AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE**

**WYDZIAŁ INFORMATYKI, ELEKTRONIKI I TELEKOMUNIKACJI**

**INSTYTUT ELEKTRONIKI**

**PROJEKT DYPLOMOWY**

*Sterownik silnika spalinowego o zapłonie iskrowym*

*Petrol engine controller*

Autor: *Mateusz Mróz*

Kierunek studiów: *Elektronika*

Opiekun pracy: *dr inż. Jacek Ostrowski*

Kraków, rok akadem. 2021/2022



Uprzedzony o odpowiedzialności karnej na podstawie art. 115 ust. 1 i 2 ustawy z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (t.j. Dz.U. z 2006 r. Nr 90, poz. 631 z późn. zm.): „Kto przywłaszcza sobie autorstwo albo wprowadza w błąd co do autorstwa całości lub części cudzego utworu albo artystycznego wykonania, podlega grzywnie, karze ograniczenia wolności albo pozbawienia wolności do lat 3. Tej samej karze podlega, kto rozpowszechnia bez podania nazwiska lub pseudonimu twórcy cudzy utwór w wersji oryginalnej albo w postaci opracowania, artystyczne wykonanie albo publicznie znieksztalca taki utwór, artystyczne wykonanie, fonogram, videogram lub nadanie.”, a także uprzedzony o odpowiedzialności dyscyplinarnej na podstawie art. 211 ust. 1 ustawy z dnia 27 lipca 2005 r. Prawo o szkolnictwie wyższym (t.j. Dz. U. z 2012 r. poz. 572, z późn. zm.) „Za naruszenie przepisów obowiązujących w uczelni oraz za czyny uchybiające godności studenta student ponosi odpowiedzialność dyscyplinarną przed komisją dyscyplinarną albo przed sądem koleżeńskim samorządu studenckiego, zwanym dalej „sądem koleżeńskim”, oświadczam, że niniejszą pracę dyplomową wykonałem(-am) osobiście i samodzielnie i że nie korzystałem(-am) ze źródeł innych niż wymienione w pracy.

## **Spis treści**

<b>WSTĘP .....</b>	<b>6</b>
<b>CELE PRACY .....</b>	<b>7</b>
<b>ROZDZIAŁ 1 OPIS TECHNOLOGII.....</b>	<b>8</b>
1.1    PODSTAWOWE INFORMACJE O SILNIKU SPALINOWYM .....	8
1.2    WYMAGANIA UKŁADU PALIWOWEGO .....	10
1.3    WYMAGANIA UKŁADU ZAPŁONOWEGO .....	13
1.4    HISTORYCZNE SYSTEMY STEROWANIA SILNIKAMI .....	14
1.4.1 <i>Układ paliwowy</i> .....	14
1.4.2 <i>Układ zapłonowy</i> .....	16
1.5    NOWOCZESNE SYSTEMY STEROWANIA SILNIKAMI .....	17
1.6    ALGORYTMY WYZNACZANIA OBCIĄŻENIA SILNIKA.....	19
<b>ROZDZIAŁ 2 PLATFORMA SPRZĘTOWA .....</b>	<b>20</b>
2.1    SILNIK .....	20
2.2    MIKROKONTROLER .....	25
2.3    CZUJNIKI I UKŁADY ELEKTRONICZNE.....	26
2.3.1 <i>Czujnik MAP</i> .....	26
2.3.2 <i>Czujniki temperatury</i> .....	27
2.3.3 <i>Czujnik położenia wału korbowego</i> .....	28
2.3.4 <i>Cewki zapłonowe</i> .....	31
2.3.5 <i>Wtryskiwacze paliwa</i> .....	32
<b>ROZDZIAŁ 3 OPROGRAMOWANIE .....</b>	<b>33</b>
3.1    ŚRODOWISKO PROGRAMISTYCZNE .....	33
3.2    BUDOWA KODU .....	35
3.2.1 <i>Biblioteka SWO</i> .....	38
3.2.2 <i>Tabele</i> .....	39
3.2.3 <i>Czujniki</i> .....	40
3.2.4 <i>Dekoder czujnika położenia</i> .....	41
3.2.5 <i>Sterowanie zapłonem</i> .....	44
3.2.6 <i>Sterowanie wtryskiem paliwa</i> .....	46
3.2.7 <i>Algorytm Speed Density</i> .....	46
3.2.8 <i>Logika działania</i> .....	48
<b>ROZDZIAŁ 4 WERYFIKACJA DZIAŁANIA PROTOTYPU .....</b>	<b>52</b>
4.1    TESTY W ŚRODOWISKU KONTROLOWANYM.....	52
4.2    TESTY NA PLATFORMIE SPRZĘTOWEJ .....	54

## **Spis treści**

---

<b>PODSUMOWANIE I WNIOSKI .....</b>	<b>56</b>
<b>BIBLIOGRAFIA .....</b>	<b>57</b>
<b>SPIS ILUSTRACJI.....</b>	<b>59</b>
<b>SPIS TABEL .....</b>	<b>61</b>

## Wstęp

Praca porusza problematykę sterowania silnikami spalinowymi wykorzystywanymi w przemyśle motoryzacyjnym, transportowym, lotniczym czy ciężkim. Ograniczono się do rozważań układów zasilanych benzyną, czyli takich o zapłonie iskrowym. W celu zapewnienia optymalnych warunków pracy silnika, niezbędne jest precyzyjne dawkowanie paliwa oraz zapewnienie zapłonu w określonym momencie. Wartości te zależą od wielu parametrów pracy silnika, które muszą być na bieżąco monitorowane.

Na przestrzeni lat technika sterowania silnikami spalinowymi rozwinęła się od prostych gaźników i aparatów zapłonowych działających czysto mechanicznie, aż do skomplikowanych systemów elektronicznych, potrafiących precyzyjnie zapewnić idealne warunki pracy silnika, samodzielnie diagnozować usterki, dostosowywać się do pojawiających się warunków pracy i wiele innych.

Geneza pomysłu na temat pracy związaną jest z fascynacją samochodami osobowymi. Branża motoryzacyjna charakteryzuje się tym, że spotykają się w niej różne dziedziny nauk m.in. akustyka, automatyka, inżynieria chemiczna, elektryczna, materiałowa, mechaniczna i wiele innych. Można pokusić się o stwierdzenie, że samochód osobowy jest dziełem sztuki.

## Cele pracy

Celem pracy jest wykonanie mikroprocesorowego sterownika silnika spalinowego o zapłonie iskrowym pozwalającym na:

- sekwencyjne dawkowanie paliwa w systemie MPI (*Multi Point Injection*) – każdy cylinder posiada swój wtryskiwacz, który sterowany jest niezależnie od pozostałych,
- sekwencyjny zapłon z użyciem jednej cewki zapłonowej na każdy cylinder, kontrolowanej bezpośrednio przez sterownik,
- możliwość współpracy z wysokoobrotowymi silnikami (optymalizacja kodu pod kątem szybkości wykonywania).

W celu przeprowadzenia testów i oceny skuteczności działania sterownika, powinna zostać przygotowana platforma sprzętowa zawierająca ramę z silnikiem oraz wszystkie potrzebne elementy pozwalające na uruchomienie i pracę silnika spalinowego.

Celem pracy jest również poznanie narzędzi wykorzystywanych w procesie tworzenia aplikacji systemów wbudowanych (ang. *toolchain*) oraz napisanie przejrzystego i czytelnego kodu.

Praca podzielona jest na kilka rozdziałów. Zawierają one kolejno:

- Rozdział 1 – prezentację zagadnień technicznych związanych z działaniem silników opisywanych w pracy, historyczne oraz nowoczesne systemy sterowania silnikami.
- Rozdział 2 – część sprzętową pracy, opis wykonania stanowiska testowego, wyboru mikrokontrolera oraz wykorzystanych czujników wraz z elektronicznymi układami kondycjonowania.
- Rozdział 3 – część związaną z oprogramowaniem, środowisko programistyczne, budowę kodu, zasadę działania sterownika.
- Rozdział 4 – testy działania prototypu silnika spalinowego.

## Rozdział 1

### Opis technologii

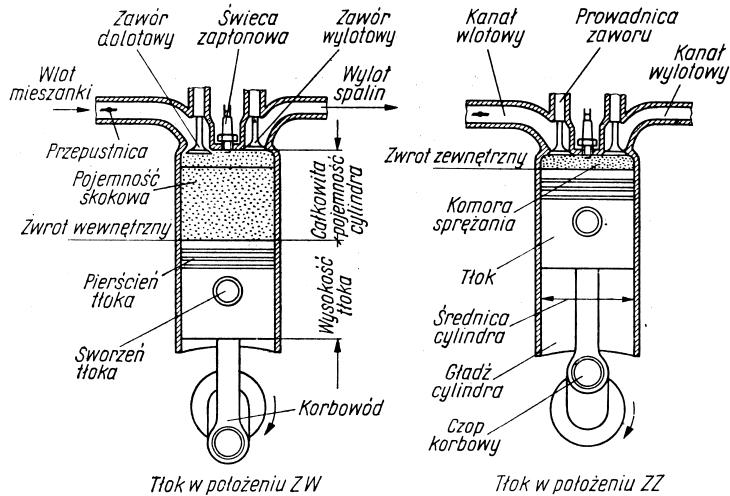
Na przestrzeni ostatnich dziesięcioleci, technika konstrukcji silników spalinowych znacząco się zmieniła. Pomimo tego zasady działania jak i problemy, z którymi muszą mierzyć się inżynierowie pozostają takie same. Zostaną one pokrótce przedstawione w tym rozdziale.

#### ***1.1 Podstawowe informacje o silniku spalinowym***

W celu przybliżenia pojęcia silnika, powołano się na książkę Wiesława Jeżewskiego *abc samochodowego silnika czterosuwowego*:

„Silnik jest to maszyna, która dostarcza energii mechanicznej kosztem doprowadzonej do niej energii innego rodzaju. (...) W silnikach o spalaniu wewnętrzny, nazywanych również silnikami spalinowymi, proces spalania paliwa odbywa się wewnątrz silnika. Zamiana energii cieplnej na mechaniczną następuje w samym silniku, a czynnikiem roboczym są spaliny. Silniki spalinowe dzielą się na tłokowe, turbinowe, odrzutowe” [1]. Detonacja paliwa może odbywać się samoczynnie lub za pomocą zapłonu elektrycznego, takie silniki nazywane są silnikami o zapłonie iskrowym. „Cykl procesów fizyko-chemicznych, powtarzających się stale w tej samej kolejności w cylindrze silnika, nazywa się cyklem pracy.” [1]. Wyróżnia się silniki dwusuwowe i czterosuwowe. Temat ten zostanie szerzej rozwinięty w dalszej części tego rozdziału. Praca skupia się tylko na tłokowych, czterosuwowych silnikach spalinowych o zapłonie iskrowym, ze względu na ich popularność w przemyśle motoryzacyjnym.

W celu wizualizacji budowy komory spalania, oraz elementów z jakich się składa, na rys. 1 przedstawiono jej przekrój poprzeczny. Tłok połączony jest przegubowo z wałem korbowym za pomocą korbowodu. Podczas pracy porusza się on wzduż gładzi cylindra, pierścienie tłokowe zapewniają szczelność komory sprężania. Elementy pozwalające na dostarczenie świeżej mieszanki paliwa z powietrzem do komory spalania, oraz opuszczenie jej przez spaliny, nazywane są zaworami. Świeca zapłonowa pozwala na pojawienie się w komorze spalania iskry, zapalającej paliwo.

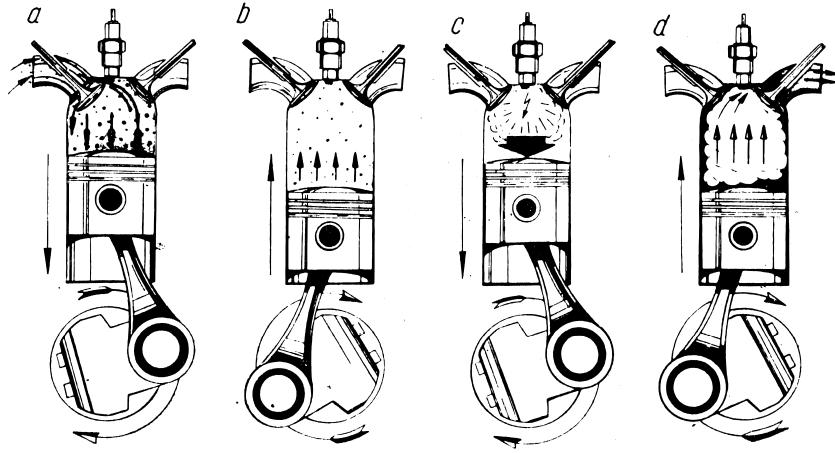


Rys. 1 Charakterystyczne położenie tłoka [1]

Przekrój komory spalania na rys. 1 przedstawia ponadto tłok w dwóch charakterystycznych pozycjach. Są to:

- ZW – Zwrot Wewnętrzny lub DMP – Dolny Martwy Punkt (*ang. BDC – Bottom Dead Center*) czyli skrajne dolne położenie tłoka w cylindrze.
- ZZ – Zwrot Zewnętrzny lub GMP – Górnny Martwy Punkt (*ang. TDC – Top Dead Center*) czyli skrajne górne położenie tłoka w cylindrze.

Ruch tłoka opisywany jest najczęściej jako przemieszczenie się od jednego z charakterystycznych punktów, do drugiego.



a — dolot, b — sprężanie, c — spalanie i rozprężanie, d — wylot

Rys. 2 Schemat działania czterosuwowego silnika o zapłonie iskrowym [1]

Zasada działania czterosuwowego silnika spalinowego o zapłonie iskrowym oraz jego cykle pracy zostały przedstawione na rys. 2. Wyróżnia są kolejno:

- a) Suw dolotu: tłok porusza się od GMP do DMP, ruch ten powoduje napływ przez otwarty zawór dolotowy świeżej mieszanki paliwa z powietrzem.
- b) Suw sprężania: tłok porusza się od DMP do GMP, w tym czasie obydwa zawory są zamknięte, przez co zassana mieszanka zostaje poddana kompresji.
- c) Suw pracy: iskra powstająca na elektrodach świecy powoduje zapłon mieszanki, powstałe w skutek tego gazy o wysokim ciśnieniu i temperaturze działając na tłok wypychając go w kierunku DMP. Obydwa zawory są zamknięte.
- d) Suw wylotu: tłok porusza się od DMP do GMP, ruch ten powoduje wypchnięcie pozostałych po spalaniu gazów przez otwarty zawór wylotowy.

Wał korbowy wykonuje więc w czasie trwania jednego cyklu (czterech suwów) obrót o  $720^\circ$ . Wał rozrządu, którego celem jest otwieranie zaworów, w czasie trwania jednego cyklu wykonuje obrót o  $360^\circ$ .

## 1.2 Wymagania układu paliwowego

W czasie trwania suwu dolotu, do komory spalania zassana zostaje mieszanka paliwa i powietrza. Ich stosunek ma wpływ na pracę silnika, jego niezawodność, moc, emisję spalin i musi być precyzyjnie dawkowany.

W opisywanych w tej pracy silnikach, rolę paliwa pełni benzyna. Teoretyczna ilość powietrza niezbędna do pełnego spalenia benzyny wynosi  $14,7 \text{ kg powietrza / kg paliwa}$ . Mieszankę taką nazywamy stochiometryczną [3]. Parametrem opisującym stosunek mas powietrza i paliwa jest AFR (ang. *Air-Fuel Ratio*) i może być opisany wzorem:

$$AFR = \frac{m_{pow}}{m_{pal}} \quad (1)$$

gdzie:

$m_{pow}$  – masa powietrza,

$m_{pal}$  – masa paliwa.

## Opis technologii

Innym parametrem służącym do określenia składu mieszanki, jest współczynnik nadmiaru powietrza  $\alpha$ , który jest równy stosunkowi masy powietrza w mieszanicy biorącej udział w spalaniu do masy teoretycznie potrzebnej do zupełnego spalania:

$$\alpha = \frac{m_{pow}}{l_o \cdot m_{pal}} \quad (2)$$

gdzie:

$m_{pow}$  – masa powietrza,

$m_{pal}$  – masa paliwa,

$l_o$  – stosunek powietrza do danego paliwa dla teoretycznej mieszanki stochiometrycznej.

Parametr  $\alpha$  jest więc unormowaną wielkością i pozwala na opis niezależny od rodzaju wykorzystywanego paliwa. Wyróżnia się 3 zakresy wartości tego parametru:

- $\alpha = 1$  – mieszanka stochiometryczna,
- $\alpha > 1$  – mieszanka zawiera więcej powietrza niż paliwa, nazywana jest mieszanką ubogą (w paliwo),
- $\alpha < 1$  – mieszanka zawiera więcej paliwa niż powietrza, nazywana jest mieszanką bogatą (w paliwo).

Granica zapalności mieszanki zależy głównie od temperatury, kilka przykładowych wartości zostało zebranych w Tabeli 1. Jak widać, wraz ze wzrostem temperatury zwiększa się zakres zapalności.

**Tabela 1 Wpływ temperatury na graniczne wartości współczynnika  $\alpha$  dla benzyny [3]**

Temperatura mieszanki [°C]	Współczynnik nadmiaru powietrza	
	Granica dolna	Granica górna
0	0,53	1,23
100	0,40	1,60
250	0,40	1,67

W wyniku spalania mieszanki stochiometrycznej powstaje dwutlenek węgla oraz woda, dla mieszanki ubogiej oprócz produktów pełnego spalania pojawia się tlen i jego zawartość rośnie wraz ze wzrostem współczynnika  $\alpha$ . Przy mieszance bogatej pojawiają się produkty niecałkowitego spalania czyli tlenek węgla i wodór [3]. Jak widać z punktu widzenia emisji szkodliwych substancji, mieszanka uboga jest pożądana.

## Opis technologii

---

Maksymalna moc silnika uzyskiwana jest dla mieszanek bogatych. Wartość współczynnika  $\alpha$  w tym przypadku zależy od rodzaju silnika oraz budowy komory spalania i wachodzi w granicach  $0,85 \div 0,95$ . Najmniejsze jednostkowe zużycie paliwa, uzyskiwane jest dla mieszanek ubogich, na ogólnie  $1,10 \div 1,20$  [10].

Silniki samochodowe pracują w szerokim zakresie obciążen, prędkości obrotowych (stałych i zmiennych), warunków atmosferycznych itd. Wszystkie te czynniki wymagają korekcji aktualnej wartości współczynnika nadmiaru powietrza z jakim pracuje silnik, aby zapewnić elastyczność i poprawną pracę dla danych warunków.

Rozruch silnika, zwłaszcza zimnego, powoduje znaczne trudności. Niska temperatura utrudnia odparowanie paliwa (co skutkuje zubożoną mieszanką), zagęszcza olej i zmniejsza pojemność akumulatora (co powoduje zmniejszenie prędkości obrotowej w trakcie rozruchu). Silnik musi być w takiej sytuacji zasilany znacznie bogatszą mieszanką, pozwalającą zmieszczenie się w granicy zapalności (Tabela 1). Im niższa jest temperatura, tym większa ilość paliwa potrzebna jest do umożliwienia rozruchu.

Bieg jałowy charakteryzuje się niskimi prędkościami obrotowymi i niewielką ilością zasysanej mieszanki. Powoduje to zmniejszenie temperatury i ciśnienia w komorze spalania, co prowadzi do zmniejszenia zakresu zapalności. Kolejnym czynnikiem utrudniającym pracę jest obciążenie silnika np. przez alternator, sprężarkę klimatyzacji czy pompę wspomagania układu kierowniczego. Wszystkie te czynniki powodują, że mieszanka powinna być wzbogacona.

Małe i średnie obciążenia, czyli jazda przy częściowo otwartej przepustnicy, jest najczęstszym zakresem pracy silnika. Ze względu na ekonomiczność i emisję spalin, mieszanka powinna być zubożona.

Wysokie obciążenie (pełne otwarcie przepustnicy) wymaga dostarczenia pełnej mocy silnika, wymusza to zastosowanie bogatej mieszanki.

Wpływ wszystkich przedstawionych stanów pracy silnika na pożądaną mieszankę zależy ponadto od aktualnej prędkości obrotowej. Jednoznaczne określenie potrzebnego wydatku paliwa jest bardzo trudne ze względu na wiele czynników. Z tego powodu dokładne wyznaczanie charakterystyk współczynnika nadmiaru paliwa odbywa się na stanowiskach pomiarowych nazywanych hamowniami, gdzie istnieje możliwość kontrolowanego obciążenia silnika i stałego monitorowania jego parametrów.

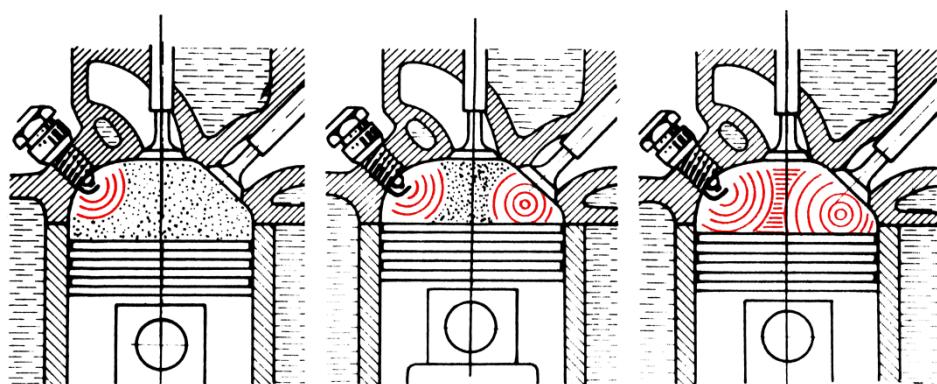
### 1.3 Wymagania układu zapłonowego

Pod koniec suwu sprężania, w celu zapalenia mieszanki, musi pojawić się iskra. Moment detonacji podawany jest w kątach obrotu wału korbowego przed GMP (*ang. BTDC – Before TDC*) i nazywany jest kątem zapłonu. Dokładna wartość kąta zapłonu zależy od wielu czynników i ma znaczący wpływ na działanie silnika.

Podstawowym powodem konieczności zmiany kąta zapłonu dla różnych warunków pracy silnika, jest fakt skończonej prędkości spalania paliwa. Wartość ta zależy m.in. od budowy komory spalania, współczynnika nadmiaru powietrza mieszanki, jej temperatury i ciśnienia. Dodatkowo wraz ze wzrostem prędkości obrotowej silnika, zapłon powinien następować wcześniej, aby pozwolić mieszance spalić się w określonym oknie czasowym [12].

Zagadnienie doboru kąta zapłonu dla różnych parametrów pracy silnika jest skomplikowane i nie będzie dokładnie poruszane w tej pracy. Należy jednak zaznaczyć, że nieprawidłowo dobrana wartość może skutkować nadmiernym grzaniem się silnika, zwiększeniem vibracji, utratą mocy lub nawet uszkodzeniem.

W tym miejscu należy krótko przybliżyć pojęcie spalania stukowego. Jest to niepożądane zjawisko samozapłonu części mieszanki. Powoduje to znaczne przeciążenia i może spowodować uszkodzenie silnika. Punktem, w którym często następuje samozapłon są zawory wylotowe, wynika to z ich wysokiej temperatury spowodowanej gorącymi spalinami. Czynnikami mogącymi zwiększyć prawdopodobieństwo wystąpienia spalania stukowego są m.in.: zbyt uboga mieszanka powodująca zwiększenie temperatury w komorze spalania, zbyt duże ciśnienie w komorze spalania, zbyt wcześnie kąt zapłonu, przegrzanie się silnika. Przykładowa wizualizacja przebiegu spalania stukowego została przedstawiona na rys. 3.



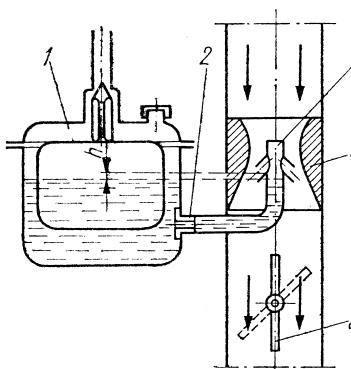
Rys. 3 Przebieg spalania stukowego [2]

## 1.4 Historyczne systemy sterowania silnikami

Pierwotnie wykorzystywane systemy sterowania silnikami działały czysto mechanicznie tzn. wykorzystywały zjawiska i prawa fizyki w celu zapewnienia odpowiednich warunków pracy.

### 1.4.1 Układ paliwowy

Historycznie najdłużej wykorzystywanym urządzeniem wytwarzającym mieszankę paliwa i powietrza w silnikach benzynowych, jest gaźnik. Został on wynaleziony w 1876 roku i był wykorzystywany w samochodach osobowych do lat 90-tych XX wieku. Konstrukcja gaźnika elementarnego (uproszczony model przedstawiający zasadę działania) została przedstawiona na rys. 4.



1 — komora pływkowa, 2 — dysza,  
3 — rozpylacz, 4 — gardziel,  
5 — przepustnica

Rys. 4 Schemat gaźnika elementarnego [3]

Gaźnik jest częścią układu dolotowego silnika, kanał wlotowy (rys. 1) znajduje się poniżej przepustnicy. Zasada działania gaźnika może być opisana równaniem Bernoulliego [4]:

$$p + \frac{1}{2} \rho v^2 + \rho gy = const. \quad (3)$$

gdzie:

- p – ciśnienie substancji w danym punkcie,
- $\rho$  – gęstość substancji,
- v – prędkość substancji w danym punkcie,
- g – przyśpieszenie grawitacyjne,
- y – wysokość w układzie odniesienia.

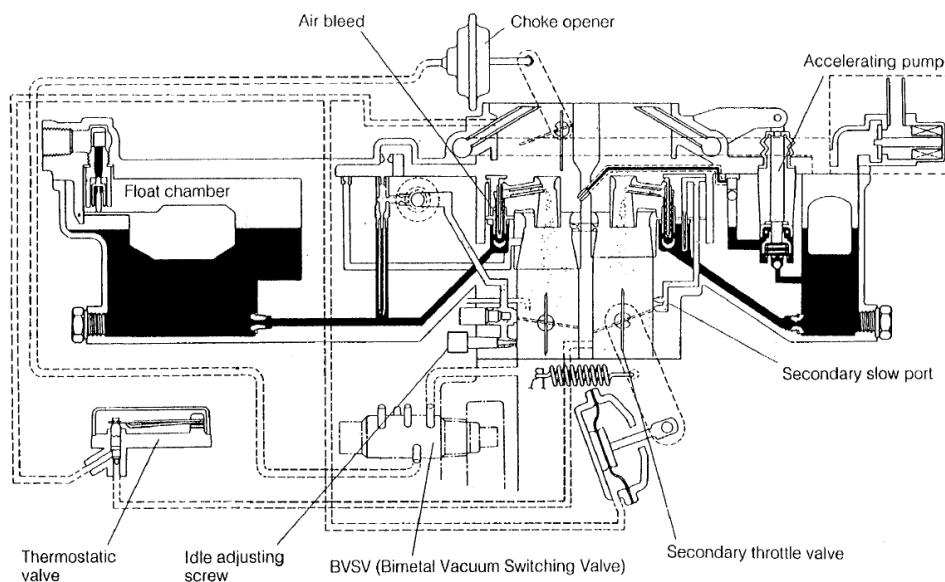
## Opis technologii

Na podstawie równania (3) dla pewnych dwóch punktów znajdujących się powyżej gardzieli (indeks 1) oraz w jej środku (indeks 2), można po krótkich przekształceniach i pominięciu wysokości zapisać:

$$p_1 + \frac{\rho}{2} v_1^2 = p_2 + \frac{\rho}{2} v_2^2 \quad (4)$$

Ze względu na zwężenie w gardzieli prawdziwa jest nierówność:  $v_2 > v_1$ , w związku z tym:  $p_2 < p_1$ . W komorze pływakowej panuje takie samo ciśnienie jak w punkcie powyżej gardzieli. Na znajdujące się tam paliwo oddziałuje więc różnica ciśnień, która powoduje przepływ paliwa przez dyszę do rozpylacza.

Przedstawione w podrozdziale 1.2 wymagania stawiane układowi paliwowemu powodują, że gaźnik elementarny nie nadaje się bezpośrednio do zasilania silników spalinowych. Musi on zostać rozbudowany o szereg mechanizmów przybliżających rzeczywistą charakterystykę gaźnika do idealnej. Powoduje to znaczne skomplikowanie konstrukcji. W celu przedstawienia praktycznego przykładu, na rys. 5 pokazano schemat ideowy gaźnika samochodu Daihatsu Charade G11 Turbo.



Rys. 5 Schemat ideowy gaźnika samochodu Daihatsu Charade G11 turbo [5]

Przedstawiony gaźnik oprócz konstrukcji z rys. 4, został wyposażony w m.in.: mechanizm wzbogacenia mieszanki w czasie rozruchu (tzw. ssanie), dyszę wolnych obrotów, drugą gardziel dostarczającą paliwo w czasie wysokich obciążeń z osobną dyszą (w tym przypadku w momencie pojawiения się ciśnienia doładowania), kanały emulsyjne spieniające paliwo przed wydostaniem się z rozpylacza.

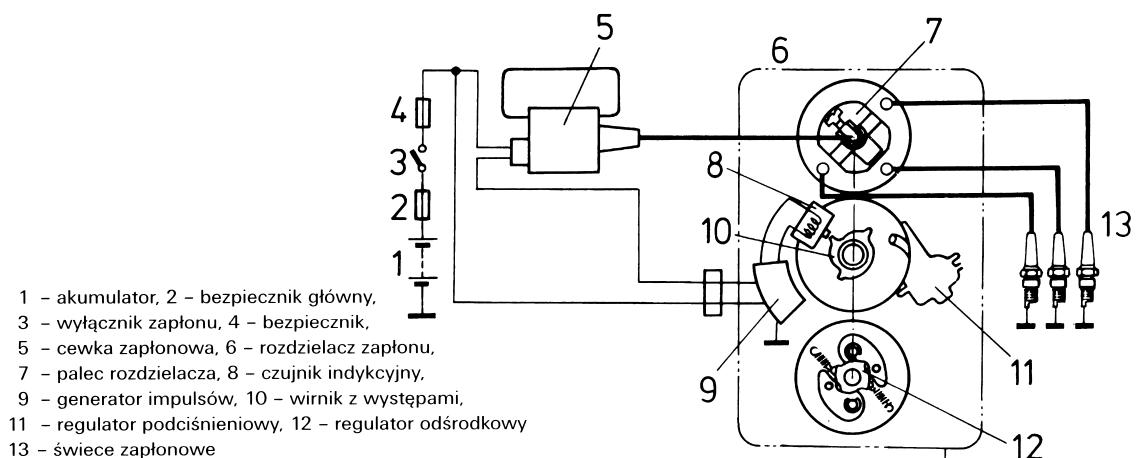
## Opis technologii

Gaźniki posiadają szereg wad, które powodują, że nie są one obecnie wykorzystywane w układach paliwowych samochodów osobowych. Należą do nich m.in.:

- Siły fizyki, m.in. bezwładność cieczy, uniemożliwia zbliżenie się do pożąданej charakterystyki układu paliwowego.
- Wymagana do działania gaźnika gardziel, efektywnie zmniejsza moc silnika poprzez utrudnianie napełniania komory spalania powietrzem. Zbyt szeroka gardziel uniemożliwia jednak silnikowi pracę przy niskich prędkościach, wymusza to stosowanie kilku gardzieli o różnych średnicach, lub zastosowanie gaźników o zmiennej szerokości gardzieli (ang. *Constant Velocity*).
- Słaba atomizacja paliwa, tzn. wydostawanie się z rozpylacza dużych kropel trudnych do odparowania. Wielkość kropel maleje w przybliżeniu z kwadratem prędkości przepływu powietrza przez gardziel [3].
- Powyższe podpunkty uniemożliwiają sprostaniu nowoczesnym, ścisłym normom emisji spalin.
- Gaźniki stały się bardzo rozbudowanymi, skomplikowanymi i precyzyjnymi urządzeniami, co powoduje trudności w wykryciu i usunięciu usterek.

### 1.4.2 Układ zapłonowy

Przykładowy schemat ideowy układu zapłonowego został przedstawiony na rys. 6.



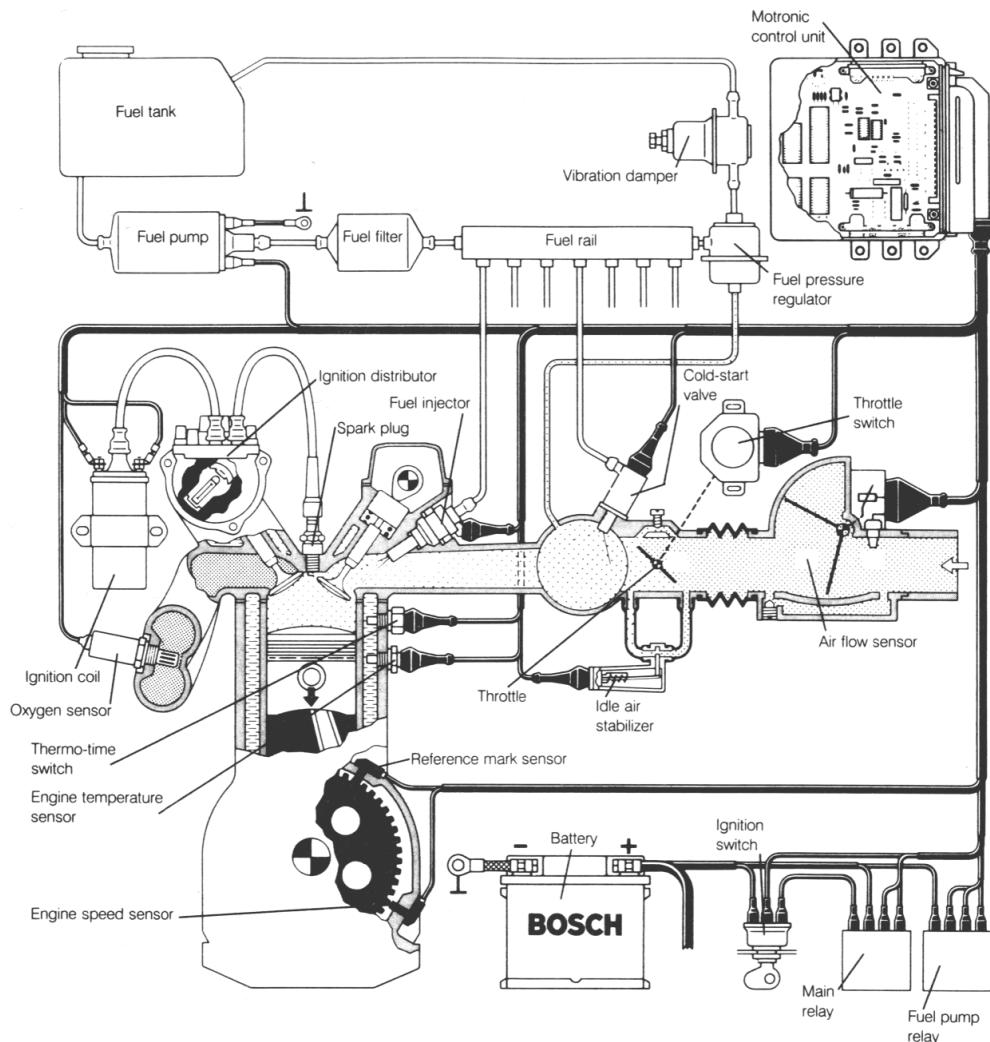
Rys. 6 Schemat ideowy układu zapłonowego samochodu Daewoo Tico [6]

## Opis technologii

Moduł rozdzielacza zapłonu napędzany jest z wału rozrządu, co zapewnia synchronizację z cyklem pracy silnika. W momentach wyznaczanych przez występy wirnika, wysokie napięcie powstające na wyjściu cewki zapłonowej zwierane jest do odpowiedniej świecy poprzez palec rozdzielacza. Kąt zapłonu korygowany jest poprzez:

- Regulator odśrodkowy: mechanizm składający się z ciężarków i sprężyn, służy do przyśpieszania momentu zapłonu wraz ze wzrostem prędkości obrotowej silnika.
  - Regulator podciśnieniowy: mechanizm przepony podłączony do kanału wlotowego silnika, opóźnia on zapłon dla wysokich podciśnień (zamkniętej przepustnicy) [10].

## **1.5 Nowoczesne systemy sterowania silnikami**



Rys. 7 Schemat nowoczesnego systemu sterowania silnikiem firmy Bosch [7]

## Opis technologii

---

Obecnie rolą obsługi nowoczesnych silników spalinowych, została całkowicie przejęta przez elektroniczne sterowniki. Ich szybkość i precyzaja praktycznie całkowicie wyeliminowały wady przedstawionych wcześniej rozwiązań. Przykładowy system został przedstawiony na rys. 7.

Do najczęściej spotykanych elementów układu elektronicznego sterowania silnikiem należą m.in.:

- Czujnik położenia wału korbowego i wału rozrządu – służą do określenia prędkości obrotowej silnika, oraz położenia poszczególnych tłoków.
- Czujnik ciśnienia w kolektorze dolotowym (*ang. MAP Manifold Absolute Pressure*) – pozwala na określenie ciśnienia napełniającego powietrzem komorę spalania.
- Przepływomierz (*ang. MAF Mass Air Flow*) – czujnik określający masę przepływającego przez niego powietrza.
- Czujniki temperatury: płynu chłodzącego (*ang. CLT Coolant Temperature*) oraz zasysanego powietrza (*ang. IAT Intake Ait Temperature*) – wykorzystywane są do korekcji ilości potrzebnego silnikowi paliwa.
- Sonda lambda: służy do określania ilości tlenu w spalinach wydechowych, na podstawie czego można wyznaczyć współczynnik nadmiaru powietrza – umożliwia korekcję dawki paliwa.
- Układ paliwowy: wysokociśnieniowa pompa, regulator ciśnienia oraz wtryskiwacze. Sterownik precyzyjnie podaje odmierzoną dawkę paliwa dla każdego cylindra osobno, wysokie ciśnienie (2.2 od 10 [bar] [8]) – umożliwia dobrą atomizację paliwa, a dzięki temu dobre zmieszanie mieszanki.
- Układ zapłonowy: sterownik silnika ładuje cewkę zapłonową, z której sygnał podawany jest na konkretny cylinder przez rozdzielacz zapłonu jak w 1.4.2. W nowszych rozwiązańach na każdy cylinder przypada osobna cewka zapłonowa, co eliminuje wszystkie elementy mechaniczne w układzie zapłonowym. Pozwala to na uzyskanie dowolnego kąta zapłonu w funkcji parametrów pracy silnika.
- Czujnik położenia przepustnicy – służy do ustalenia kąta otwarcia przepustnicy.
- Silnik krokowy biegu jałowego (*ang. IAC Idle Air Controller*) – służy do korekcji prędkości biegu jałowego, zmieniającej się pod wpływem obciążenia silnika.

## 1.6 Algorytmy wyznaczania obciążenia silnika

Algorytmy wyznaczania obciążenia silnika wykorzystywane są przez sterownik do obliczenia zapotrzebowania silnika na paliwo, za pomocą danych odczytyanych z czujników. Należą do nich m.in. [11]:

- Algorytm Speed Density: obciążenie wyznaczane jest na podstawie masy powietrza trafiającej do komory spalania. Obliczana jest ona z wykorzystaniem ciśnienia w kolektorze dolotowym oraz temperatury powietrza. Algorytm ten może być stosowany zarówno w silnikach wolnossących jak i turbodoładowanych.
- Algorytm wykorzystujący czujnik MAF: obciążenie wyznaczane jest na podstawie masy powietrza trafiającej do komory spalania, odczytanej wprost z czujnika, co zmniejsza liczbę potrzebnych do wykonania operacji matematycznych. Algorytm ten może być stosowany zarówno w silnikach wolnossących jak i turbodoładowanych.
- Algorytm Alpha-N: obciążenie wyznaczane jest na podstawie kąta otwarcia przepustnicy. Może być zastosowane tylko w silnikach wolnossących. Znajduje zastosowanie w przypadku braku stabilnego źródła ciśnienia, np. w silnikach z pojedynczą przepustnicą na każdy cylinder (*ang. ITB Individual Throttle Bodies*).

W wielu algorytmach wyznaczania obciążenia silnika spalinowego, pod uwagę brana jest sprawność napełniania cylindra (*ang. VE Volumetric Efficiency*). Wyznaczana jest ona jako stosunek masy mieszanki znajdującej się w komorze spalania po suwie dolotu, do teoretycznej masy mieszanki, która znalazłaby się w komorze spalania, gdyby gęstość powietrza znajdująca się w cylindrze była równa atmosferycznej [9]. VE zależy od obciążenia silnika oraz prędkości obrotowej. Czynnikami zmniejszającymi sprawność napełniania cylindra są elementy układu wlotowego powodujące spadek ciśnienia, np. filtr powietrza, przepustnica czy kanały wlotowe głowicy. Parametr ten może zostać zwiększyony poprzez zastosowanie turbodoładowania, lub wykorzystania zjawisk rezonansowych w układzie wlotowym.

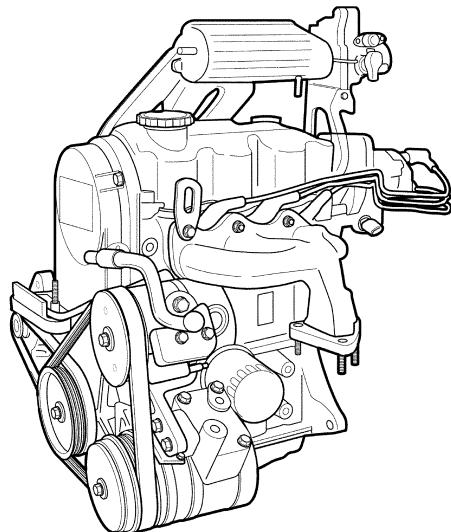
## Rozdział 2

### Platforma sprzętowa

Platforma sprzętowa składa się z części mechanicznej czyli stanowiska testowego zawierającego silnik spalinowy wraz z niezbędnymi do jego poprawnego działania elementami, oraz elektronicznej, zawierającej prototyp sterownika silnika, potrzebne do jego działania układy elektroniczne oraz czujniki i aktuatora.

#### 2.1 Silnik

Jako obiekt badawczy, wybrany został silnik marki Daewoo o oznaczeniu F8CV, montowany w modelu Matiz w latach 1997 – 2004. Przedstawiony został on na rys. 8.



Rys. 8 Silnik Daewoo F8CV [13]

Wybór tego silnika podyktowany został jego popularnością, niską ceną, dostępnością części, niewielkim rozmiarem, wagą, prostotą konstrukcji oraz fabrycznym wyposażeniem w szereg czujników wykorzystywanych w systemie elektronicznego sterowania silnikiem.

Jest to trzy cylindrowy, czterosuwowy silnik spalinowy z zapłonem iskrowym o pojemności  $796 \text{ [cm}^3\text{]}$ . Posiada dwa zawory na każdy cylinder oraz chłodzenie cieczą. Jest to więc przykład silnika, którego zasady pracy zostały opisane w Rozdzia³ 1.

Ze względu na długość skrzyni biegów, oraz braku zastosowania w projekcie dla znajdujących się w niej przekładni, została ona oddzielona od obudowy tarczy sprzęgłowej, znajdującej się bezpośrednio przy bloku silnika. Zdemontowana została również tarcza sprzęgła oraz docisk. Kolejnym etapem była fabrykacja zaślepki, która została przykręcona do części obudowy tarczy sprzęgłowej, gdzie fabrycznie przykręcana jest skrzynia biegów. Pozwoliło to na wykorzystanie przykręcanej do niej rozrusznika oraz zabezpieczenie poruszającego się z dużą prędkością koła zamachowego. Dodatkowo zaślepka umożliwiła wykonanie w tym miejscu mocowania silnika do ramy nośnej. Czynności te pozwoliły na zmniejszenie rozmiarów oraz masy konstrukcji, poprzez wyeliminowanie niepotrzebnych elementów, z równoczesnym zachowaniem wymaganych funkcjonalności. Etapy prac zostały przedstawione na rys. 9.



Rys. 9 Prace przy obudowie tarczy sprzęgłowej

Aby zapewnić silnikowi możliwość oddawania ciepła w czasie pracy, stanowisko zostało wyposażone w układ chłodzenia: chłodnica płynu, zbiornik wyrównawczy oraz przewody chłodnicze. Przygotowany silnik został przedstawiony na rys. 10. Wymagana była również modyfikacja elastycznych przewodów płynu chłodniczego, ze względu na inne niż w pojeździe umiejscowienie chłodnicy względem silnika. Zaślepiono również krótkie nagrzewnicy znajdujące się w bloku.

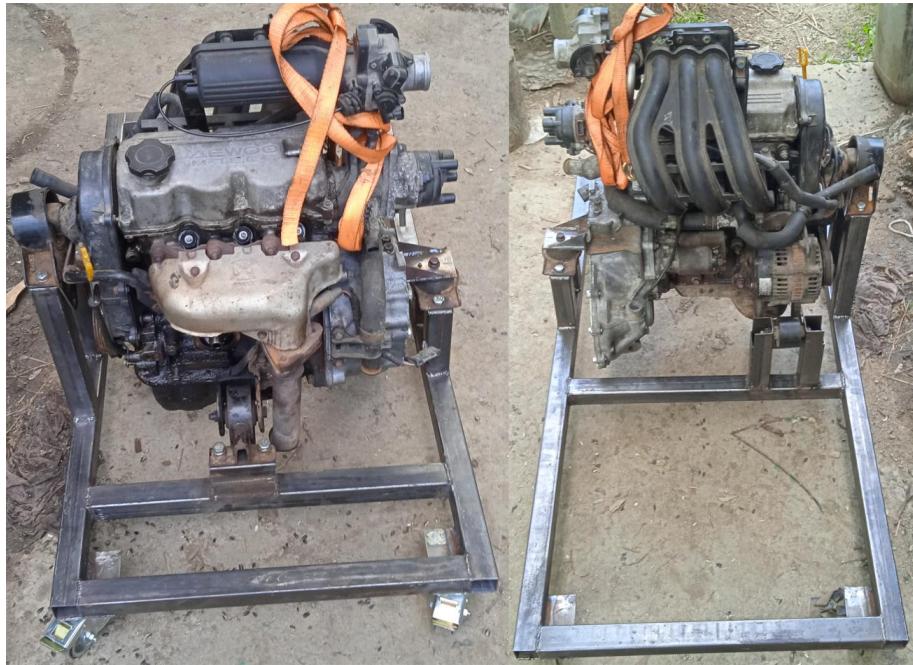


**Rys. 10 Silnik z układem chłodzenia**

Projekt ramy nośnej został wykonany w oparciu o wymiary silnika i osprzętu. Pod uwagę wzięte zostało również potrzebne miejsce na wydech oraz ewentualne przyszłe modyfikacje konstrukcji. Zdecydowano się na wykorzystanie oryginalnych mocowań silnika w konstrukcji samochodu, są to 4 gumowo – metalowe łączniki zapewniające tłumienie drgań powstałych w czasie pracy. Miejsca łączenia silnika z ramą zostały wykonane tak, aby zapewnić prostopadłość silnika względem podłoża. Rama została wykonana z zamkniętych profili stalowych 40x40[mm] w celu zapewnienia sztywności i wytrzymałości konstrukcji. Spoiny wykonano przy pomocy metody MAG. Aby zapewnić stanowisku możliwość przemieszczania się, rama została wyposażona w koła. Na rys. 11 oraz rys. 12 przedstawiono wykonaną ramę oraz przymocowany do niej silnik.



**Rys. 11 Rama nośna silnika**



**Rys. 12 Silnik przymocowany do ramy**

Elementy układu chłodzenia, zostały zamocowane w ramie jak pokazano na rys. 13. Zbiornik wyrównawczy znajduje się w najwyższym punkcie układu aby mógł spełniać swoją rolę. Chłodnicę płynu połączono z ramą nośną poprzez elementy gumowe. Pozwala to na zabezpieczenie jej przed uszkodzeniem mechanicznym, spowodowanym drganiami. Wszystkie elementy posiadają możliwość łatwego demontażu.



**Rys. 13 Układ chłodzenia zamontowany na ramie**

Kolejnym elementem niezbędnym do zapewnienia stanowisku pomiarowemu możliwości pracy, jest układ wydechowy. Wykorzystane zostały elementy oryginalne, poddane jednak drobnym modyfikacjom. Układ wydechowy składa się z kolektora wydechowego, łącznika oraz tłumików, modułowa konstrukcja pozwala na łatwy demontaż. Kolejne elementy skręcane są ze sobą za pomocą fabrykowanych kryz oraz uszczelniane materiałem odpornym termicznie. Łącznik posiada część elastyczną, zaś wydech zawieszony jest na ramie dwoma łącznikami gumowymi. Zapewnia to izolację układu wydechowego od drgań powstałynych przez pracę silnika, oraz zabezpieczenie przed uszkodzeniem. Wykorzystano dwa tłumiki, dzięki czemu zmniejszona została emisja hałasu. Zdjęcia układu wydechowego zamontowanego na ramie zostały przedstawione na rys. 14.



Rys. 14 Układ wydechowy

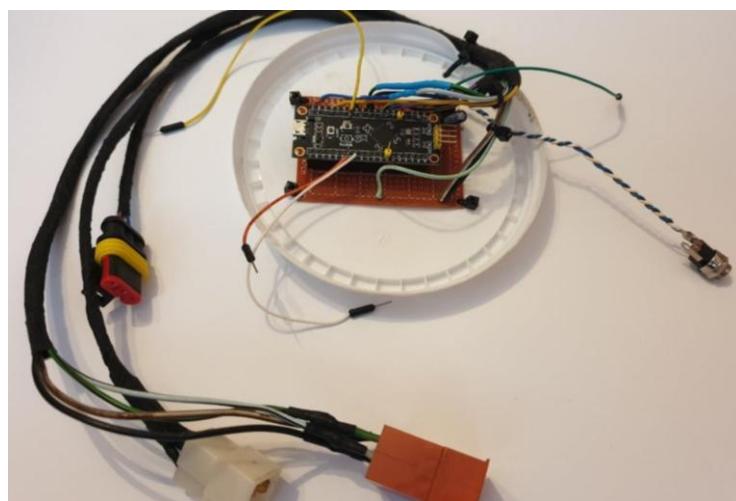
Wykonano ponadto dwie przegrody, w jednej znajduje się akumulator i jest ona przymocowana do obudowy osłony tarczy sprzęgłowej. Druga, w której znajduje się zbiornik z paliwem, jest umiejscowiona na ramie, przy złączu listwy paliwowej. Rama w celu ochrony przed korozją, została pomalowana czarną farbą epoksydową, zaś wydech farbą dedykowaną do tego celu, odporną na temperaturę.

## 2.2 Mikrokontroler

Wykorzystany mikrokontroler to STM32F411CE. W trakcie jego wyboru, wzięte zostały pod uwagę wymagania stawiane mikrokontrolerowi w aplikacji sterowania silnikami spalinowymi. Dodatkowo pamiętano o założeniach projektu, zawartych w Cele pracy. Najważniejsze parametry mikrokontrolera [15]:

- Maksymalna częstotliwość taktowania rdzenia procesora: 100[MHz]. Możliwość bezzwłocznego wykonywania instrukcji z pamięci Flash (*ART Accelerator*). Pozwala to na szybkie wykonywanie kolejnych instrukcji, a przez to na możliwość obsługi wysokoobrotowych silników.
- Rdzeń 32-bit ARM Cortex-M4 z koprocessorem zmienoprzecinkowym FPU oraz z instrukcjami DSP. Pozwala na szybkie wykonywanie operacji na liczbach zmienoprzecinkowych, które wykorzystywane są np. w algorytmach wyznaczania obciążenia silnika (rozdział 1.6). Dodatkowo wykorzystany rdzeń umożliwia *debuggowanie* czyli monitorowanie jego pracy oraz wykrywanie i usuwanie błędów w pisany kodzie.
- Peryferia liczników (*timers*): sześć 16 – bitowych oraz dwa 32 – bitowe z możliwością taktowania do 100[MHz]. Pozwalają na precyzyjne odliczanie czasów w szerokim zakresie.
- Układ jest tani i ogólnie dostępny (stan w momencie pisania pracy), dzięki modułowi ewaluacyjnemu firmy WeAct [16].

Prototyp oparty o płytę ewaluacyjną został przedstawiony na rys. 15.



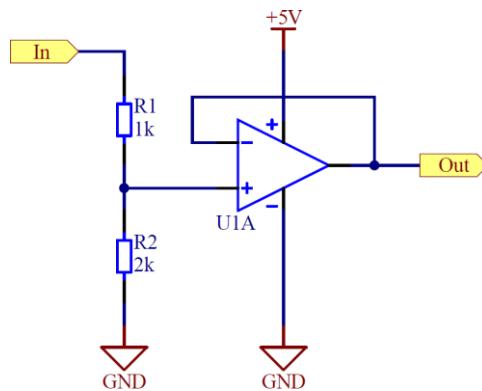
Rys. 15 Prototyp sterownika silnika spalinowego

## 2.3 Czujniki i układy elektroniczne

Wykorzystane w pracy czujniki pochodzą, poza cewkami zapłonowymi, z fabrycznego wyposażenia silnika. W tym podrozdziale zostaną one opisane.

### 2.3.1 Czujnik MAP

Jest to czujnik aktywny, generujący sygnał napięciowy w funkcji ciśnienia. Charakterystyka sensora jest liniowa w zakresie pracy, który wynosi 15 – 102 [kPa] [14]. Czujnik zasilany jest napięciem 5[V] i takie też napięcie może pojawić się na jego wyjściu. Powoduje to problem z bezpośrednim podłączeniem go do wejścia mikrokontrolera. Jego przetwornik ADC wykorzystuje napięcie odniesienia wynoszące 3,3[V], co wymusza redukcję napięcia wyjściowego czujnika. Schemat wykorzystanego w tym celu układu został przedstawiony na rys. 16.



Rys. 16 Schemat układu kondycjonowania sygnału z czujnika MAP

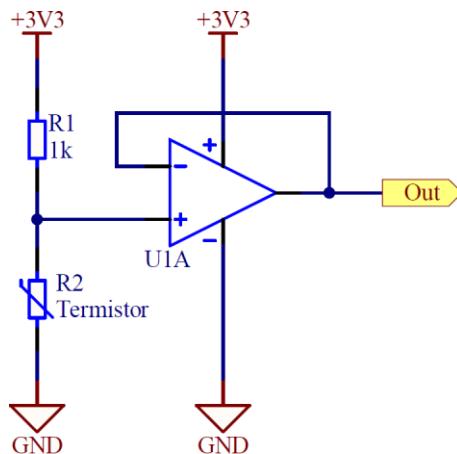
Wartości rezystancji zostały dobrane tak, aby dla maksymalnego napięcia wejściowego 5[V], napięcie wyjściowe wynosiło:

$$U_{OUT_{MAX}} = U_{IN} \cdot \frac{R_1}{R_1 + R_2} \approx 3,3[V] \quad (5)$$

Wzmacniacz operacyjny zapewnia izolację dzielnika napięcia od obciążenia. Jest to niezbędne, gdyż rezystancja obciążenia spowodowałaby zmianę współczynnika podziału czyli członu mnożonego z  $U_{IN}$  w równaniu (5). Wzmacniacz operacyjny jest zasilany napięciem 5[V], ponieważ nie jest on w stanie wysterować swojego wyjścia do wartości równej napięciu zasilania. Tak więc w przypadku zasilania z 3,3[V] niemożliwym byłoby uzyskanie tej wartości na wyjściu.

### 2.3.2 Czujniki temperatury

Czujnik temperatury płynu chłodzącego (CLT) oraz temperatury zasysanego powietrza (IAT) wykorzystują termistory NTC, czyli elementy, których rezystancja zmniejsza się wraz ze wzrostem temperatury. Wykorzystywane są one w układzie jak na rys. 17.



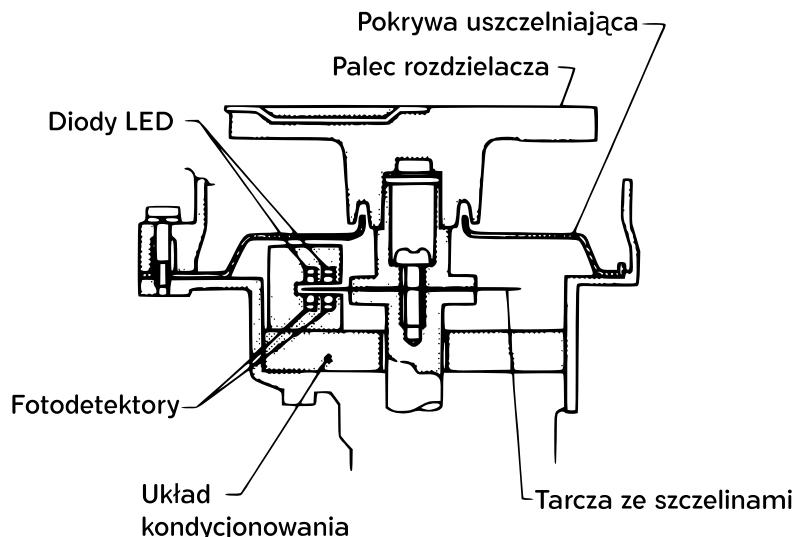
Rys. 17 Schemat układu do pomiaru temperatury

Termistor tworzy z rezystorem dzielnik napięcia, wraz ze zmianą temperatury, zmienia się rezystancja termistora, a przez to napięcie wyjściowe. Nie jest ono jednak liniowo zależne od temperatury. Wzmacniacz operacyjny tak jak na rys. 16 służy do zmniejszenia wpływu rezystancji obciążenia na współczynnik podziału dzielnika napięcia. Jednym ze sposobów konwersji sygnału uzyskanego z wyjścia układu na wartość temperatury, jest matematyczny opis tej zależności, a następnie implementacja tych działań w sterowniku. Ta metoda jest jednak czasochłonna, zamiast tego wykonano kilkanaście pomiarów napięcia wyjściowego układu dla pewnych wartości temperatury, a następnie stabelaryzowano je w kodzie sterownika.

Czujniki CLT oraz IAT zbudowane są w oparciu o ten sam termistor, różnią się jednak obudową. Wynika to z warunków środowiska pracy. Czujnik IAT musi być czuły na zmiany temperatury, i szybko oddawać zgromadzone ciepło, dlatego najczęściej obudowa wykonana jest z tworzywa sztucznego. Czujnik CLT zaś, nie ma takich wymagań, ponieważ temperatura płynu chłodniczego nie zmienia się tak szybko. Przez to jego obudowa może być bardziej wytrzymała i posiadać większą inertję, najczęściej wykonuje się ją z metali.

### 2.3.3 Czujnik położenia wału korbowego

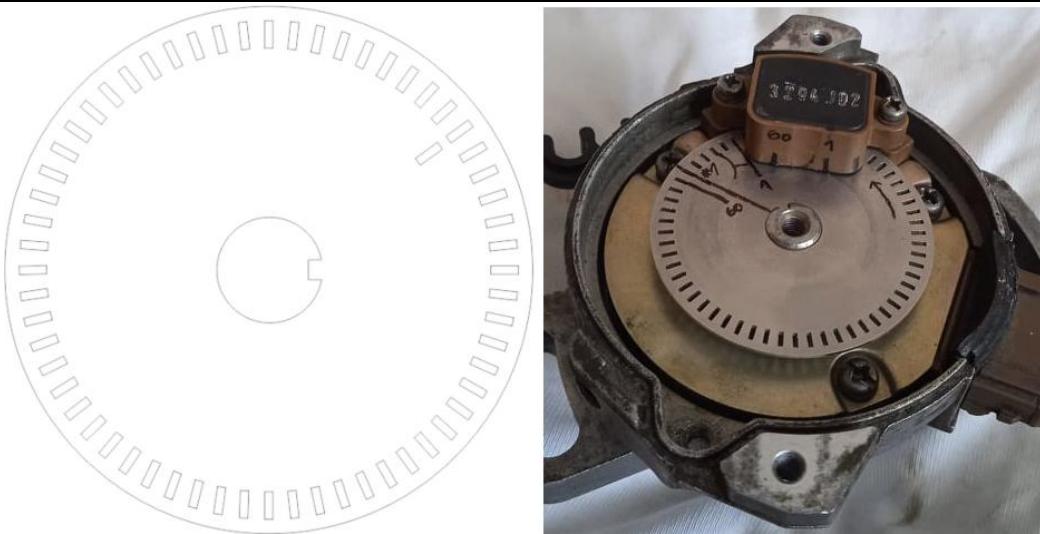
Optyczny czujnik położenia wału korbowego znajduje się w aparacie zapłonowym. Schemat takiego rozwiązania przedstawiony jest na rys. 18.



Rys. 18 Przekrój optycznego czujnika położenia silnika w aparacie zapłonowym [17]

Rotor napędzany przez wał rozrządu, oprócz dystrybucji zapłonu poprzez palec rozdzielacza (1.4.2), wprawia również w ruch tarczę ze szczelinami. Ta zaś, porusza się w szczelinie transoptora złożonego z diod LED (długość emitowanego światła znajduje się poza zakresem widzialnym), oraz fotodetektorów. Sygnał uzyskany w ten sposób jest podawany na układ kondycjonowania, który generuje sygnały wyjściowe wykorzystywane przez sterownik silnika. Aby ten system mógł niezawodnie pracować, niezbędna jest szczelność aparatu, chroniąca przed nieuniknionymi w motoryzacji zanieczyszczeniami oraz zewnętrznymi źródłami światła.

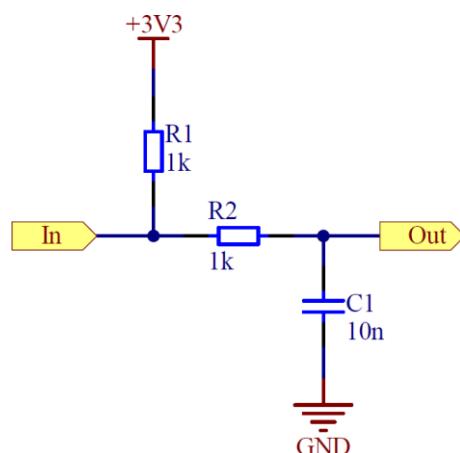
W celu uzyskania prostego do odczytu przez sterownik sygnału, zaprojektowano nową tarczę ze szczelinami, a następnie zlecono laserowe wycięcie tego kształtu w stali nierdzewnej. Schemat elementu oraz jego aplikacja zostały przedstawione na rys. 19. Nie zdecydowano się na wykorzystanie oryginalnej tarczy, ponieważ posiadała ona kolejne grupy szczelin oddzielone przerwami. W opisywanym projekcie, to rozwiązanie nie ma zastosowania. Ponadto usunięcie przerw między kolejnymi grupami szczelin zwiększa rozdzielcość otrzymywanych przez sterownik sygnałów.



**Rys. 19 Projekt tarczy oraz wykonana tarcza w aparacie zaplonowym**

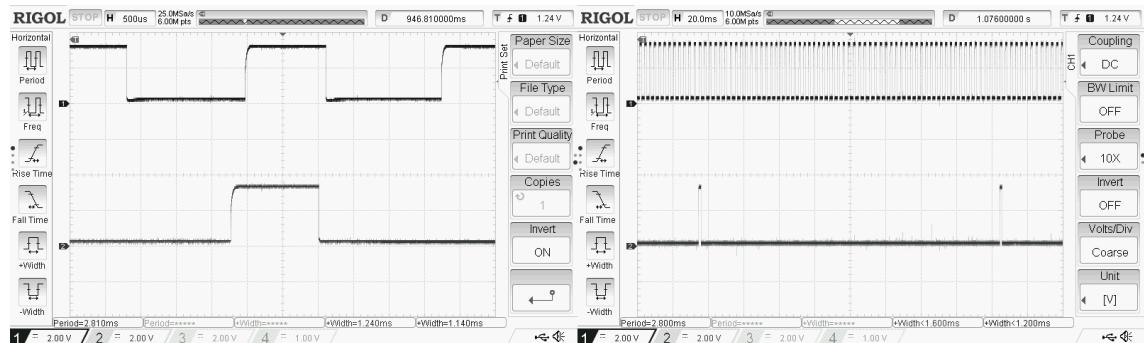
Tak jak można zauważyć na rys. 18, transoptor szczelinowy posiada parę diod i fotodetektorów, co pozwala na jednoczesny odczyt z dwóch rzędów szczelin na tarczy. Pierwszy rząd zawiera 60 otworów i służy do wykrywania prędkości obrotowej, zaś drugi, pojedynczą szczelinę, pozwalającą na odczytanie dokładnego położenia poszczególnych cylindrów.

Wyjścia czujnika położenia wału korbowego, są wyjściami typu otwarty – kolektor (dren), tzn. są albo zwarte do masy albo w stanie wysokiej impedancji. W celu zapewnienia poprawnego odczytu sygnałów przez sterownik, niezbędne jest więc zastosowanie rezystora podciągającego do linii zasilania. Dodatkowo zdecydowano się na zastosowanie pasywnego filtra dolnoprzepustowego RC pierwszego rzędu, pozwalającego na eliminację zakłóceń oraz zmianę kształtu sygnału. Schemat zaprojektowanego układu został przedstawiony na rys. 20.



**Rys. 20 Schemat układu wejściowego dla sygnału położenia**

Rezystor podciągający do zasilania posiada wartość  $1[\text{k}\Omega]$ . Jest to niewielka wartość, która pozwala jednak na bardzo szybkie przeładowanie pojemności linii, a przez to uzyskanie stromych zboczy w sygnale. Wadą tego rozwiązania jest zwiększone wydzielanie się mocy na samym rezystorze. Dolnoprzepustowy filtr pasywny, posiada 3dB częstotliwość odcienia równą ok.  $16[\text{kHz}]$  dla idealnego źródła sygnału i obciążenia układu. Uzyskane na wyjściu przebiegi, po podaniu na wejście sygnałów z czujnika położenia, zostały zaprezentowane na rys. 21.



Rys. 21 Osciylogramy przedstawiające sygnały z czujnika położenia wału korbowego

Prawy oscylogram przedstawia przebiegi z czasu trwania ponad jednego pełnego obrotu tarczy. Pojedynczy sygnał pojawia się raz na 60 sygnałów wykorzystywanych do określenia prędkości obrotowej wału korbowego, co potwierdza prawidłowy odczyt szczelin zaprojektowanej tarczy. Lewy oscylogram przedstawia odczytane sygnały w przybliżeniu. Układ z rys. 20 spełnia swoją funkcję: nie pojawiają się oscylacje sygnału w momencie przełączania, zbocza są gładkie i stromo nachylone. Odczytane z oscyloskopu parametry sygnału, zostały zebrane w Tabela 2.

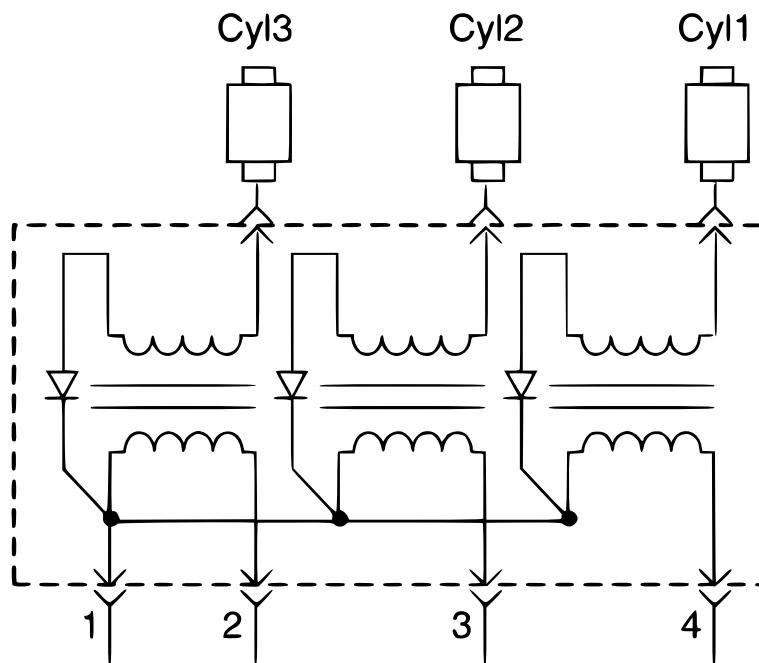
Tabela 2 Parametry zboczy sygnału położenia wału korbowego

Parametr	Zbocze sygnału	
	Narastające	Opadające
Czas trwania (10%-90%)	19[μs]	170[ns]

Jak więc widać, sygnały te mogą być wykorzystane do poprawnego wyzwalania przerwań mikrokontrolera.

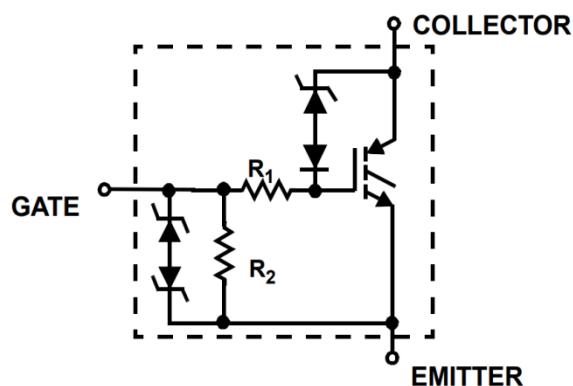
### 2.3.4 Cewki zapłonowe

Zdecydowano się na niewykorzystanie fabrycznego układu zapłonowego, w celu eliminacji podatnych na usterki elementów mechanicznych i zwiększeniu precyzji momentu zapłonu. Aparat zapłonowy służy więc tylko do generacji sygnałów położenia silnika. Wykorzystano zestaw cewek zapłonowych z nowszego modelu samochodu Matiz, ich schemat został zaprezentowany na rys. 22.



Rys. 22 Schemat cewek zapłonowych [13]

Na każdy cylinder przypada jedna cewka, sterownik silnika steruje bezpośrednio ładowaniem każdej z nich. Elementem kluczującym jest tranzystor IGBT o oznaczeniu ISL9V2540 firmy Fairchild Semiconductor pokazany na rys. 23.



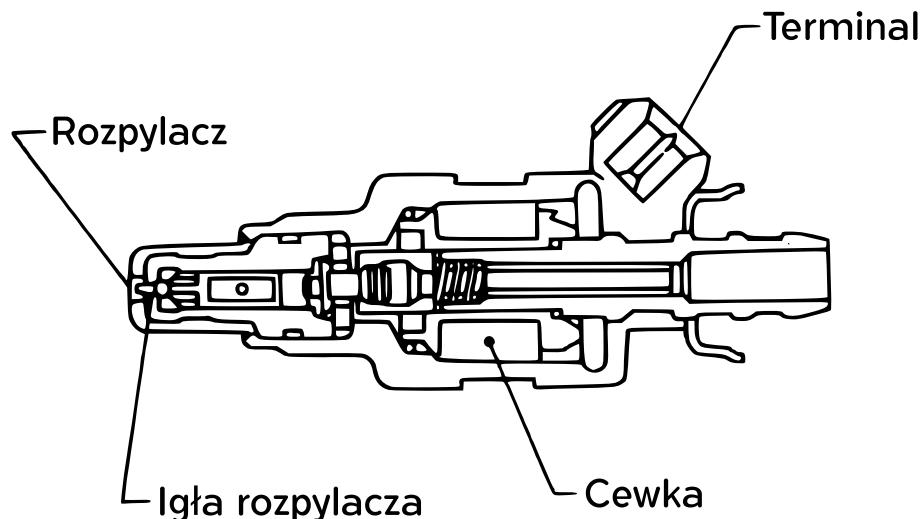
Rys. 23 Symbol tranzystora IGBT ISL9V2540 [18]

Jest to układ przystosowany do sterowania cewkami zapłonowymi, może pracować z pojawiającymi się w układzie wysokimi napięciami, a ponadto nie wymaga zewnętrznych elementów do działania.

Sterownik wymusza przepływ prądu przez konkretną cewkę, tak aby zmagała ona odpowiednią ilość energii, a następnie rozwiera układ ładowania. Wtedy na uzwojeniu wtórnym cewki pojawia się wysokie napięcie powodujące przeskok iskry pomiędzy elektrodami świecy i nastąpienie zapłonu. Czas ładowania cewki (*ang. coil dwell time*) zależy od parametrów samej cewki oraz układu ładowania. Dla tych wykorzystywanych w motoryzacji są to wartości pojedynczych milisekund.

### 2.3.5 Wtryskiwacze paliwa

Schemat wtryskiwacza paliwa został przedstawiony na rys. 24.



Rys. 24 Schemat wtryskiwacza paliwa [17]

W momencie przepływu prądu przez cewkę, powstające pole magnetyczne przesuwa igłę, pozwalając paliwu pod wysokim ciśnieniem wydostać się z rozpylacza. Wtryskiwacze charakteryzują się czasem martwym (*ang. Injector dead time*) tzn. czasem pomiędzy pojawieniem się napięcia na terminalu, a rozpoczęciem dostarczania paliwa, wydajnością wtryskiwacza oraz impedancją cewki. Wykorzystywane w projekcie cewki posiadają wysoką impedancję (ok.  $12[\Omega]$ ) dzięki czemu mogą być załączane prądem stałym.

Sterownik silnika podobnie jak w podrozdziale 2.3.4 steruje każdym wtryskiwaczem osobno. Jako element kluczujący wykorzystano również układ ISL9V2540.

## Rozdział 3

# Oprogramowanie

Pod pojęciem oprogramowania rozumie się narzędzia służące do rozwoju programu sterownika, generowania z niego wynikowego kodu w języku maszynowym oraz sam kod w języku C.

### 3.1 Środowisko programistyczne

Ze względu na potrzebę uzyskania szybkiego wykonywania kodu, zdecydowano się na wybór języka C oraz skorzystanie z bibliotek CMSIS. Pozwala to na odwoływanie się bezpośrednio do rejestrów procesora oraz jego peryferii. Poza zwiększeniem szybkości wykonywania kodu, daje to okazję do dokładnego poznania mechanizmów działania wykorzystywanych układów np. timerów.

Jako edytor kodu, wykorzystany został program firmy Microsoft: Visual Studio Code [19]. Jest on udostępniany na darmowej licencji MIT. Program ten umożliwia m.in.: szybkie wywoływanie zaprogramowanych wcześniej poleceń konsoli systemu operacyjnego, wiele skrótów klawiszowych i ułatwień przyśpieszających proces pisania kodu, system *IntelliSense* pozwalający na automatyczne uzupełnianie kodu w trakcie pisania. Środowisko to pozwala również na wywoływanie z jego poziomu wszystkich wykorzystywanych narzędzi programistycznych.

Do budowy projektu wykorzystano oficjalnie wspierający architekturę ARM zestaw narzędzi GNU Arm Embedded Toolchain [20] udostępniany na darmowej licencji FSF. Zawiera on linker, kompilator GCC, biblioteki oraz zestaw innych pomocnych narzędzi. Producent udostępnia też dokładną dokumentację umożliwiającą dostosowanie kompilatora do potrzeb danego projektu.

Jako narzędzie służące do programowania i debuggowania kodu wybrano OpenOCD [22] udostępniany na darmowej licencji GNU GPL v2.0. Wspiera on architekturę ARM oraz szereg standardów komunikacji z rdzeniem. Jako programator wybrano STLink v2.0, ze względu na niską cenę, oraz kompaktowy rozmiar. Komunikuje się on z mikrokontrolerem z wykorzystaniem protokołu SWD.

W celu automatyzacji budowania projektu, wykorzystano narzędzie GNU Make [21] udostępniany na darmowej licencji FSF. Potrafi on rozpoznać zależności między plikami w projekcie i w ten sposób wywołać dla każdego z nich polecenia kompilatora z odpowiednimi parametrami. Na potrzeby projektu napisano plik *makefile* zawierający instrukcje dla narzędzia. Zawiera on m.in.:

- Budowę projektu dla dwóch celów: Debug oraz Release. Wybierany jest on w zależności od podanego w momencie wywołania pliku make parametru. Wersja Debug wywołuje kompilator bez optymalizacji kodu (flaga -Og) oraz z flagami powodującymi generowanie plików umożliwiających debuggowanie (-g3 -gdwarf-2). Wersja Release wywołuje kompilator z optymalizacją kodu pod kątem prędkości wykonywania (flaga -Ofast).
- Zestaw flag kompilatora i linkera, m.in.: określające rodzaj mikrokontrolera, zestaw instrukcji asemblerowych, które zostaną wygenerowane, rozszerzenie pliku wyjściowego, optymalizacja kodu, wykorzystanie sprzętowego koprocesora liczb zmiennoprzecinkowych, rodzaje generowanych ostrzeżeń (wybrano najbardziej restrykcyjne wykrywanie możliwych problemów).
- Ścieżki do plików źródłowych, nagłówkowych oraz bibliotek dołączanych do projektu w momencie linkowania.
- Komendy konsoli systemowej pozwalającej na wyczyszczenie zawartości folderu z plikami wynikowymi oraz zaprogramowanie i wyczyszczenie pamięci flash mikrokontrolera poprzez serwer OpenOCD
- Możliwość wypisywania na ekran konsoli wywoływanych poleceń, lub zamiast tego jedynie nazwy wykonywanego procesu oraz pliku, na którym on działa. Przykład tekstu uzyskanego w ten sposób został przedstawiony na rys. 25.

W trakcie procesu rozwoju kodu, korzystano z systemu kontroli wersji Git udostępnianego na darmowej licencji GNU GPL v2.0. Pozwala on na śledzenie procesu powstawania zmian w kodzie, możliwość wygodnego udostępniania i przechowywania kopii zapasowych projektu.

```
> Executing task: make -j8 all DEBUG=1 <

Compiling file: build/system_stm32f4xx.o
Compiling file: build/engine_constants.o
Compiling file: build/engine_sensors.o
Compiling file: build/ignition_driver.o
Compiling file: build/injection_driver.o
Compiling file: build/main.o
Compiling file: build/speed_density.o
Compiling file: build/swo.o
Compiling file: build/tables.o
Compiling file: build/trigger_decoder.o
Compiling file: build/utils.o
Assembling file: build/startup_stm32f411xe.o
Linking target: build/ECU.elf
    text      data      bss      dec      hex filename
    7856        24     1592    9472    2500 build/ECU.elf
Preparing: build/ECU.hex
Preparing: build/ECU.bin
```

Rys. 25 Widok konsoli po wywołaniu budowy projektu przez narzędzie make

### 3.2 Budowa kodu

Kod napisany został w języku C z wykorzystaniem bibliotek CMSIS. Duży nacisk położony został na uzyskaniu przejrzystego i czytelnego kodu, w tym celu podjęto następujące działania. Środowisko Visual Studio Code umożliwia tworzenie szablonów plików, dzięki czemu uzyskuje się konsekwencję schematu ich budowy. Przykład pliku nagłówkowego stworzonego w taki właśnie sposób został przedstawiony na rys. 26. Można zauważać podział na sekcje logiczne opatrzone czytelnym komentarzem.

```
/*=====
 * File: common_include.h
 * Project: ECU
 * Author: Mateusz Mroz
 * Date: 13.09.2021
 * Brief: Common include
 *=====
#ifndef _COMMON_INCLUDE_H_
#define _COMMON_INCLUDE_H_

/*=====
 * INCLUDE SECTION
 *=====
#include "stm32f411xe.h"

#include "float.h"
#include "stdbool.h"
#include "stddef.h"
#include "stdint.h"
#include "utils.h"

/*=====
 * EXPORTED DEFINES AND MACRO SECTION
 *=====
#define DisableIRQ ..... (_disable_irq)
#define EnableIRQ ..... (_enable_irq)

/*=====
 * EXPORTED GLOBAL VARIABLES SECTION
 *=====
/*=====

/*=====
 * EXPORTED FUNCTION DECLARATION SECTION
 *=====
/*=====

#endif
/* end of file */
```

Rys. 26 Przykład pliku nagłówkowego stworzonego według szablonu

Wybrany został określony styl kodu i ściśle go przestrzegano. Przykład definicji oraz deklaracji funkcji napisany według niego, przedstawiony został na rys. 27. Funkcje posiadają w nazwie akronim modułu w którym zostały zdefiniowane, deklaracje funkcji posiadają dokładny opis działania oraz parametrów w postaci komentarza, zmienne lokalne korzystają z konwencji *camelCase* (kolejne wyrazy pisane są łącznie, rozpoczynając każdy następny wielką literą, z wyjątkiem pierwszego).

```
/*=====
 * brief: .....Calculate spark angle
 * param[in]: .....speed - engine speed in RPM
 * param[in]: .....pressure - engine absolute pressure in kPa
 * param[in]: .....channel - current engine channel
 * param[out]: .....None
 * return: .....float - spark fire angle
 * details: .....None
 */
static float SpDen_CalculateSpark(float speed, float pressure, EnCon_CylinderChannels_T channel);

/*=====
 * Function: SpDen_CalculateSpark
 * =====*/
static float SpDen_CalculateSpark(float speed, float pressure, EnCon_CylinderChannels_T channel)
{
    float tableAngle;

    if (SPDEN_IGNITION_ANGLE_LOCK)
    {
        tableAngle = SPDEN_LOCKED_ANGLE;
    }
    else
    {
        tableAngle = Tables_Get3DTableValue(TABLES_3D_SPARK, speed, pressure);
    }

    return UTILS_CIRCULAR_DIFFERENCE(spden_work_tdc_angles[channel], tableAngle, ENCON_ENGINE_FULL_CYCLE_ANGLE);
}
```

Rys. 27 Przykład deklaracji oraz definicji funkcji

Projekt podzielony jest na moduły, czyli najczęściej pary plików nagłówkowego i źródłowego. Każdy z nich odpowiada za pewną część funkcjonalności na co wskazuje jego nazwa. Projekt składa się z następujących modułów:

- *utils* – moduł zawiera funkcje i makra wykorzystywane w wielu miejscach, np. konwersje jednostek, przedrostków SI, działania na buforach kołowych.
- *common\_include* – moduł wykorzystywany przez wszystkie inne moduły, zawiera dołączenie do kodu bibliotek: *float*, *stdbool*, *stddef*, *stdint*, CMSIS oraz modułu *utils*.
- *engine\_constants* – moduł zawiera makra opisujące parametry silnika i jego osprzętu, a także przechowuje informacje o aktualnej prędkości silnika i położeniu tłoków.
- *swo* – moduł zawierający własną implementację funkcji *printf*, zoptymalizowaną pod kątem prędkości działania. Umożliwia ona wypisanie komunikatów (np. znaków ASCII) przez pin SWO standardu SWD.
- *timers* – moduł opisujący wykorzystanie liczników znajdujących się w mikrokontrolerze. Zapewnia on nadzór nad wykorzystaniem ich zasobów.

- *tables* – moduł zawierający tabele: trójwymiarowe i dwuwymiarowe. Pierwsze z nich to tabele VE oraz kąta zapłonu. Wartości te podawane są w funkcji ciśnienia w kolektorze dolotowym oraz prędkości obrotowej silnika. Dwuwymiarowe tabele zawierają zależności: napięcia wejściowego od temperatury czujnika CLT i IAT, a także parametru wzbogacenia mieszanki w funkcji temperatury silnika. Moduł zapewnia dodatkowo aproksymację wartości odczytywanych z dyskretnych tabel (aproksymacja liniowa oraz biliniowa).
- *engine\_sensors* – moduł wykorzystuje przetwornik ADC do mierzenia wartości napięć z wejść mikrokontrolera podłączonych do czujników (IAT, CLT i MAP). Posiada również funkcje zwracające odczytane wartości w różnych jednostkach (np. odczytanie wartości temperatury w Kelvinach).
- *trigger\_decoder* – moduł wykorzystuje mechanizm przerwań oraz sprzętowy licznik do dekodowania sygnałów otrzymywanych z czujnika położenia wału korbowego. Uzyskiwana jest w ten sposób informacja o prędkości obrotowej silnika oraz położeniu kolejnych tłoków.
- *ignition\_driver* – moduł wykorzystuje sprzętowy licznik do sterowania wyjściami mikrokontrolera odpowiedzialnymi za zapłon mieszanki.
- *injection\_driver* – moduł wykorzystuje sprzętowy licznik do sterowania wyjściami mikrokontrolera odpowiedzialnymi za wtrysk paliwa.
- *speed\_density* – moduł ten wykorzystując pozostałe moduły realizuje logikę sterowania silnikiem w algorytmie speed – density (podrozdział 1.6).
- *main* – główny moduł programu, zawiera inicjalizację pozostałych modułów oraz nieskończoną pętlę.
- *system* – moduł zapewniony przez bibliotekę CMSIS, zawiera funkcjonalności związane z zarządzaniem zegarami. Jest on wykorzystywany w obsłudze przerwania po resecie mikrokontrolera. Do pliku została dodana konfiguracja pętli synchronizacji fazy (PLL), dzięki której uzyskuje się maksymalną częstotliwość pracy rdzenia (100[MHz]) jeszcze przed wywołaniem funkcji main.

W podrozdziałach 3.2.1 do 3.2.8 zostaną opisane dokładnie poszczególne funkcjonalności kodu.

### 3.2.1 Biblioteka SWO

Biblioteka ta korzysta z ITM (ang. *Integrated Trace Macrocell*), narzędzie to pozwala na przesyłanie do niego z wysoką szybkością danych z kodu programu, a następnie enkapsulację i przeniesienie ich na port wyjściowy. Portem tym jest pin SWO (ang. *Serial Wire Output*) standardu SWD. Dodatkowo ITM posiada niewielką kolejkę FIFO, zabezpieczającą przed przepełnieniem bufora, a także nadaje każdemu pakietowi swój znacznik czasu [23].

W kodzie zaimplementowana została funkcja działająca podobnie jak *printf*, jest ona jednak znacznie prostsza, a przez to szybsza i mniejsza. Tak samo przyjmuje ona zmienną liczbę argumentów i wykorzystuje znak „%” wraz ze specjalnym markerem do formatowania tekstu wyjściowego. Obsługiwane są zmienne typu *signed*, *unsigned* (w zapisie decimalnym i heksadecimalnym) oraz *string*. Dzięki wykorzystaniu kompilatora GCC, istnieje również możliwość sprawdzenia zgodności markera z odpowiadającą jej zmienną na etapie działania preprocesora. Przykład uzyskanego w ten sposób ostrzeżenia został przedstawiony na rys. 28.

```
Core/Src/trigger_decoder.c: In function 'TIM3_IRQHandler':
Core/Src/trigger_decoder.c:204:17: warning: format '%u' expects argument of type 'unsigned int', but argument 2 has type 'double' [-Wformat=]
  204 |     Swo_Print("%u RPM \n", EnCon_GetEngineSpeed());
        |          ^~~~~~
        |          | double
        |          | unsigned int
        |          %f
```

Rys. 28 Ostrzeżenie uzyskane przez podanie zmiennej złego typu dla markera

Biblioteka zawiera również możliwość wyświetlania akronimu modułu, z którego została wywołana wraz z numerem linii. Ułatwia to rozróżnienie komunikatów w przypadku dużej ilości informacji. Biblioteka znalazła główne zastosowanie w procesie testów, do wyświetlania aktualnego stanu pracy silnika np.: prędkości obrotowej czy kąta położenia pierwszego cylindra. Przykład kodu pokazującego działanie biblioteki, wraz z uzyskanymi przez środowisko programistyczne danych z mikrokontrolera, pokazano na rys. 29.

```
/*-----*
 * Function: main
 *-----*/
int main(void)
{
    SWO_Init();

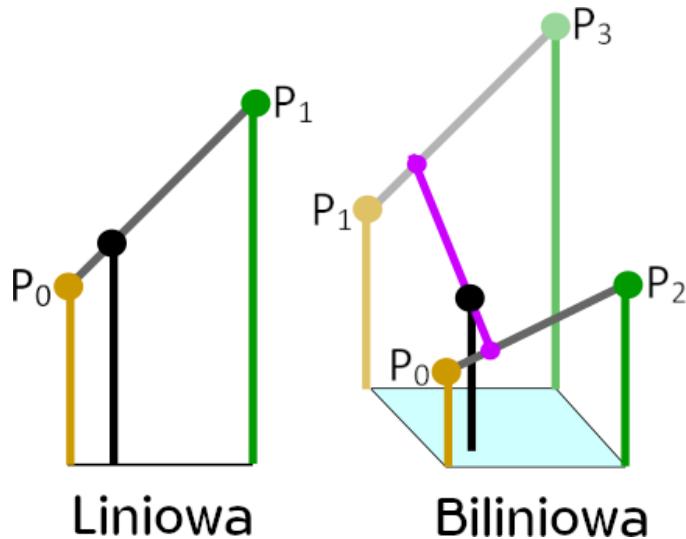
    while(1)
    {
        SWO_Print("Hello world from SWO %d \n", 1u);
        SWO_PrintLog("Hello world from SWO %d \n", 2u);
        Main_MsDelay(1000u);
    }
}
```

Hello world from SWO 1  
MAIN 79: Hello world from SWO 2  
Hello world from SWO 1  
MAIN 79: Hello world from SWO 2  
Hello world from SWO 1  
MAIN 79: Hello world from SWO 2  
Hello world from SWO 1  
MAIN 79: Hello world from SWO 2  
Hello world from SWO 1  
MAIN 79: Hello world from SWO 2  
Hello world from SWO 1  
MAIN 79: Hello world from SWO 2  
Hello world from SWO 1  
MAIN 79: Hello world from SWO 2  
Hello world from SWO 1  
MAIN 79: Hello world from SWO 2  
Hello world from SWO 1  
MAIN 79: Hello world from SWO 2  
Hello world from SWO 1  
MAIN 79: Hello world from SWO 2  
Hello world from SWO 1  
MAIN 79: Hello world from SWO 2  
Hello world from SWO 1

Rys. 29 Przykład działania biblioteki SWO

### 3.2.2 Tabele

Sterownik silnika korzysta ze stabelaryzowanych wartości, które zostały zaprogramowane w procesie konfiguracji. Ze względu na ograniczone zasoby systemu wbudowanego, tabele posiadają wartości dla 16 dyskretnych punktów na każdej z osi, co jednak dla danego zastosowania jest wystarczające. Wartości dla których istnieje potrzeba odczytania zawartości tabel, posiadają jednak o wiele większą rozdzielcość. Zazwyczaj są to zmienne zapisywane w pamięci mikrokontrolera na 32 bitach, co daje  $2^{32} - 1$  wartości. Wymusza to przybliżenie danych znajdujących się między najbliższymi punktami w tabelach. W tym celu zaimplementowano algorytm interpolacji liniowej dla tabel dwuwymiarowych, oraz interpolacji biliniowej dla trójwymiarowych. Graficzna prezentacja działania tych metod przedstawiona została na rys. 30.



Rys. 30 Porównanie metod interpolacji: liniowej i biliniowej [24]

Odczytywane wartości (czarne kropki) znajdują się na prostych poprowadzonych przez najbliższe znane punkty.

Metodę interpolacji liniowej można opisać wzorem:

$$y = \frac{y_0(x_1 - x) + y_1(x - x_0)}{x_1 - x_0} \quad (6)$$

gdzie:

$x, y$  – współrzędne szukanego punktu,

$x_0, y_0$  – współrzędne punktu  $P_0$ ,

$x_1, y_1$  – współrzędne punktu  $P_1$ ,

dla punktów oznaczonych jak na rys. 30.

Metodę interpolacji biliniowej można opisać wzorem:

$$z = (1 - u)((1 - v)z_1 + vz_2) + u((1 - v)z_3 + vz_4) \quad (7)$$

dla:

$$u = \frac{x - x_1}{x_3 - x_1} \quad (8)$$

$$v = \frac{y - y_1}{y_2 - y_1} \quad (9)$$

gdzie:

$x, y, z$  – współrzędne szukanego punktu,

$x_0, y_0, z_0$  – współrzędne punktu  $P_0$ ,

$x_1, y_1, z_1$  – współrzędne punktu  $P_1$ ,

$x_2, y_2, z_2$  – współrzędne punktu  $P_2$ ,

$x_3, y_3, z_3$  – współrzędne punktu  $P_3$ ,

dla punktów oznaczonych jak na rys. 30.

Wzory (6) i (7) zostały zaimplementowane w kodzie. Dzięki temu możliwe jest odczytanie wartości z tabel, dla dowolnej wartości znajdującej się pomiędzy minimalnymi a maksymalnymi punktami na osiach. W przypadku wykroczenia poza te zakresy, zwracana jest wartość najbliższego punktu.

### 3.2.3 Czujniki

Moduł przetwornika analogowo cyfrowego (ADC) jest wykorzystywany do odczytywania napięć z wejść mikrokontrolera. Podczas konfiguracji skupiono uwagę na uzyskaniu możliwie szybkiego działania mechanizmu, z równoczesnym uzyskaniem odpowiednio precyzyjnych pomiarów. Przetwornik może działać z maksymalną częstotliwością zegara wynoszącą 36[MHz] [15]. Magistrala APB2, do której podłączony jest moduł ADC, taktowana jest z częstotliwością 100[MHz]. Jednak wejściowy dzielnik częstotliwości zegara dla przetwornika, może przyjmować wartości 1, 2, 4, ..., co umożliwia uzyskanie maksymalnie wartości 25[MHz]. Wybrana rozdzielcość przetwornika wynosi 12 bitów, wymagany czas konwersji napięcia wejściowego wynosi dla tego przypadku 15 cykli zegara ADC. Mniejsze rozdzielcości umożliwiają szybszą konwersję, jednak różnica ta wynosi dla np. 8 bitów zaledwie 4 cykle zegara, co dla wybranej częstotliwości pracy odpowiada czasowi 160[ns].

Rozdzielcość zaś wynosi odpowiednio 4095 oraz 255 wartości, co argumentuje wybór. Dla każdego kanału można skonfigurować dodatkowy czas konwersji, wynosi on minimalnie 3 cykle zegara i taka wartość została właśnie wybrana ze względu na to, że nie ma ona wpływu na precyzję działania modułu ADC [25].

Wykorzystywany jest bezpośredni dostęp do pamięci (DMA) wraz z trybem ciągłej konwersji przetwornika ADC (*scan mode*), wyzwalanej przez kod programu i zaprzestającej pracę po obsłudze wszystkich kanałów (*single conversion mode*). Umożliwia to maksymalnie szybkie wykonanie wszystkich potrzebnych pomiarów, bez potrzeby interwencji z poziomu kodu programu. Przetwornik po zakończeniu konwersji z danego kanału, wysyła rządzenie DMA, które wpisuje odczytaną wartość do pamięci RAM mikrokontrolera. W tym czasie moduł ADC wykonuje kolejny pomiar. Ten zestaw czynności wykonywany jest dla każdego czujnika, a po zakończeniu takiej sekwencji przetwornik zatrzymuje się i generowane jest przerwanie [25].

Całkowity czas trwania sekwencji pomiarów, może być wyznaczony za pomocą wzoru:

$$T_{conv} = x \left( \frac{n_{base} + n_{add}}{f_{ADC}} \right) \quad (10)$$

gdzie:

$T_{conv}$  – całkowity czas trwania sekwencji pomiarów,

$x$  – liczba kanałów,

$n_{base}$  – czas konwersji dla danej rozdzielcości w cyklach zegara,

$n_{add}$  – dodatkowy czas konwersji w cyklach zegara,

$f_{ADC}$  – częstotliwość zegara taktującego moduł ADC.

Dla opisanej konfiguracji oraz trzech wykorzystywanych czujników (MAP, CLT, IAT), czas ten wynosi 2[ $\mu$ s].

### 3.2.4 Dekoder czujnika położenia

Jak zostało to opisane w rozdziale 2.3.3, informacje o położeniu silnika pochodzą z tarczy, posiadającej dwa rzędy szczelin, w jednej znajduje się ich 60, a w drugiej jedna. Porusza się ona z prędkością wału rozrządu co jest wyjątkowo przydatne, ponieważ silniki czterosuwowe charakteryzują się tym, że na jeden cykl pracy (cztery suwy), przypadają dwa obroty wału korbowego, czyli jeden obrót wału rozrządu. Dzięki temu,

że pojedynczy sygnał pojawia się raz w czasie trwania 4 suwów, służy on do odczytania aktualnego położenia tłoków.

Prędkość obrotowa silnika podawana jest w obrotach wału korbowego na minutę (RPM). Do odczytania tej wartości służy 60 szczelin znajdujących się na tarczy. Położenie tłoków opisywane jest za pomocą kąta położenia pierwszego z nich. Może on przyjmować wartości od  $0^\circ$  do  $720^\circ$ , gdzie wartość  $0^\circ$  oznacza GMP suwu kompresji. Określenie położenia pozostałych tłoków jest proste, ponieważ są one przesunięte względem siebie o stały kąt, w przypadku trzycylindrowego silnika wykorzystanego w pracy wartość ta wynosi  $120^\circ$ . Należy jednak zauważyć, że kąty GMP suwu kompresji dla kolejnych cylindrów są przesunięte względem siebie o  $240^\circ$ . Dodatkowo zapłon następuje najpierw dla pierwszego cylindra, potem dla trzeciego, a na końcu dla drugiego. Pozwala to na minimalizację drgań silnika.

Do wyznaczenia prędkości obrotowej wału korbowego, wykorzystany został 16 bitowy licznik, w trybie wyczwalania od sygnału wejściowego (*input capture*). Taktowany jest on częstotliwością  $1[\text{MHz}]$ , co pozwala na uzyskanie rozdzielczości  $1[\mu\text{s}]$ . Częstotliwość ta jest uzyskana dzięki podzieleniu częstotliwości magistrali zegara wynoszącej  $100[\text{MHz}]$  przez 100. Licznik posiada skonfigurowane przerwanie od przepełnienia rejestru liczącego, oraz od pojawienia się na wejściu mikrokontrolera opadającego zbocza. W momencie przerwania od zbocza, zapisywany jest obecny stan rejestru liczącego i jeżeli posiadana jest prawidłowa ostatnia wartość tego rejestru, przez ich różnicę można obliczyć czas, który upłynął. Pojawienie się dwóch przerwań od przepełnienia rejestru liczącego, oznacza niższą niż minimalną, możliwą do wykrycia prędkość obrotową silnika i jest to traktowane jako rozsynchronizowanie się sterownika z silnikiem. Prędkość obrotowa silnika w RPM wyznaczana jest ze wzoru:

$$n = \frac{60}{x_{imp} \frac{\Delta_{reg}}{f_{tim}}} \quad (11)$$

gdzie:

$n$  – prędkość obrotowa silnika [RPM]

$x_{imp}$  – liczba impulsów z czujnika położenia w trakcie jednego obrotu wału

$\Delta_{reg}$  – różnica między wartościami rejestrów licznika

$f_{tim}$  – częstotliwość taktowania modułu licznika

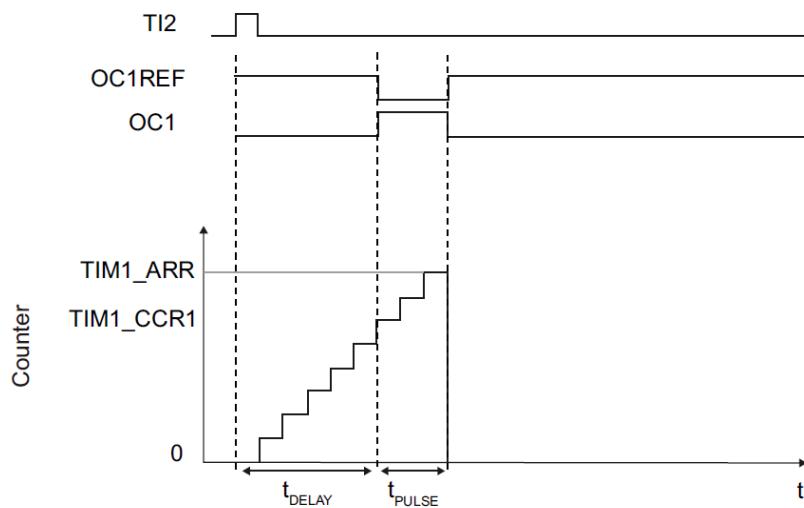
Dla wykorzystywanej w projekcie tarczy wartość  $x_{imp}$  wynosi 30. Można łatwo obliczyć, że minimalna prędkość silnika jaką można wykryć to ok. 30[RPM]. Teoretycznie maksymalną możliwą do wykrycia prędkością jest 2.000.000[RPM], co jest oczywiście obecnie wartością nieosiągalną dla silników motoryzacyjnych. W celu ograniczenia ilości instrukcji wykonywanych w obsłudze przerwania licznika, prędkość obrotowa w RPM nie jest tam wyliczana. ISR zapisuje tylko różnicę wartości rejestrów liczących. Gdy następuje potrzeba odczytania prędkości silnika, wartość ta jest jednorazowo wyliczana i zapamiętywana do momentu pojawienia się nowej danej z przerwania. Dodatkowo możliwe jest odczytanie prędkości w obrotach na minutę, jak również jako różnicę wartości rejestrów liczących.

Sterownik posiada zaprogramowane przerwanie od zbocza narastającego na wejściu podłączonym do sygnału synchronizacji (pojawiającego się raz na cykl pracy). Gdy zdarzenie to zostanie zarejestrowane, przy kolejnym przerwaniu od zbocza opadającego na wejściu liczącym prędkość obrotową silnika, następuje synchronizacja sterownika z silnikiem. Oznacza to, że znane jest aktualne położenie wału korbowego. Sterownik posiada zaprogramowaną wartość kąta silnika jaka ma miejsce w momencie synchronizacji, wynika ona z położenia pojedynczej szczeliny na tarczy. Gdy synchronizacja się powiodła, każde kolejne przerwanie licznika od opadającego zbocza sygnału prędkości silnika, zwiększa wartość kąta położenia o odpowiednią wartość. Wynika ona z ilości szczelin tarczy, w projekcie jest ich 60 i odpowiadają one  $720^\circ$  obrotu wału korbowego, więc kąt inkrementowany jest o  $12^\circ$ . W momencie pojawienia się kolejnego sygnału synchronizacji sprawdzane jest, czy wyliczone wartości pokrywają się.

Należy zwrócić uwagę, że sterownik generuje przerwania od różnych zboczy sygnałów sterujących. Dzięki temu istnieje mniejsze prawdopodobieństwo, wystąpienia przerwania, zanim obsługa poprzedniego jeszcze się nie skończyła. Mogłoby to skutkować nieprzewidzianym zachowaniem kodu, co jest niedozwolone. Dodatkowo, przerwanie od licznika prędkości obrotowej, posiada największy priorytet w projekcie. Dzięki temu nawet w przypadku wystąpienia innego, będzie ono oczekiwane na skończenie tego o najwyższym priorytecie [23].

### 3.2.5 Sterowanie zapłonem

Sterowanie cewkami zapłonowymi odbywa się za pomocą licznika w trybie porównania sterującego pinem wyjściowym (*output capture/compare*) oraz pojedynczego wyzwalania (*one-shot*). Zasada działania tak skonfigurowanego licznika została przedstawiona na rys. 31.



Rys. 31 Zasada działania licznika sterującego cewką zapłonową [25]

Licznik załączany jest z poziomu kodu programu. Zawartość rejestru liczącego porównywana jest w każdym cyklu zegara z zaprogramowaną wartością (CCR – *Capture/Compare Register*). Gdy wartości te zrównają się ze sobą, następuje zmiana stanu pinu wyjściowego na wysoki (OC – *Output Compare*). Licznik dalej inkrementuje wartość rejestru liczącego, aż do osiągnięcia zaprogramowanej maksymalnej wartości (ARR – *Auto Reload Register*). W tym momencie stan pinu wyjściowego zostaje zmieniony z powrotem na niski, rejestrów są zerowane, a licznik zostaje zatrzymany.

Działając w ten sposób, możliwe jest takie zaprogramowanie licznika, aby po pewnym czasie rozpoczął ładowanie cewki i w określonym momencie spowodował powstanie iskry zapłonowej. Dzięki takiemu działaniu kod programu wylicza jednorazowo potrzebne wartości, w znanym momencie wyznaczonym przez przerwania od sygnału prędkości obrotowej rozpoczęta działanie licznika i nie musi w danym cyklu wykonywać już żadnych operacji z tym związanych.

Wybrany licznik posiada 32 bitowy rejestr liczący i jest taktowany z maksymalną częstotliwością 100[MHz]. Pozwala to na odliczanie czasów od 10[ns] do ok. 43[s] z rozdzielcością 10[ns], co zapewnia dużą precyzję oraz szeroki zakres czasów pracy.

Wartości rejestrów, które należy wyznaczyć w celu zaprogramowania licznika zaprezentowano na wzorach [25]:

$$ARR = t_{delay} + t_{pulse} - 1 \quad (12)$$

$$CCR = ARR + 1 - t_{pulse} \quad (13)$$

gdzie:

ARR – maksymalna wartość rejestru liczącego,

CCR – wartość rejestru liczącego, dla którego następuje zmiana stanu wyjściowego,

$t_{pulse}$  – czas ładowania cewki,

$t_{delay}$  – czas od rozpoczęcia działania licznika do rozpoczęcia ładowania cewki.

W momencie obliczania parametrów, znany jest kąt silnika w jakim zostanie rozpoczęte działanie licznika oraz kąt dla którego ma nastąpić zapłon. Dla tych danych wzór (12) można przekształcić do postaci:

$$ARR = \frac{\Delta_\alpha}{\alpha_{trig}} m \Delta_{n\_reg} \quad (14)$$

gdzie:

$\Delta_\alpha$  – różnica między kątami silnika dla rozpoczęcia licznika oraz nastąpienia zapłonu,

$\alpha_{trig}$  – kąt silnika odpowiadający jednemu sygnałowi położenia,

m – stosunek częstotliwości zegarów: licznika położenia i licznika zapłonu ,

$\Delta_{n\_reg}$  – prędkość silnika podana w różnicy między wartościami rejestrów liczących licznika położenia.

Wykonanie obliczeń ze wzoru (14) przez mikrokontroler jest bardzo szybkie, gdyż korzysta on bezpośrednio z wartości rejestrów liczących reprezentujących prędkość silnika oraz składa się z iloczynu stałej i dwóch zmiennych. Równanie (13) jest realizowane jako suma wartości innego rejestru i stałej, co również zabiera bardzo mało czasu.

Kąt zapłonu zależy od prędkości obrotowej silnika oraz ciśnienia w kolektorze dolotowym, dlatego też jego zależność jest zapisana w pamięci programu jako trójwymiarowa mapa.

### 3.2.6 Sterowanie wtryskiem paliwa

Sterowanie wtryskiem paliwa odbywa się za pomocą takiego samego 32 bitowego licznika taktowanego zegarem 100[MHz] jak w przypadku sterowania cewkami zapłonowymi (2.3.4). Działa on również w takim samym trybie. Po pewnym czasie opóźnienia, pin wyjściowy przyjmuje stan wysoki na dokładnie określony czas, potrzebny do dostarczenia precyzyjnej dawki paliwa.

Różnica pojawia się w sposobie wyliczania wartości potrzebnych rejestrów. W momencie rozpoczęcia obliczeń, znany jest kąt silnika w jakim zostanie rozpoczęte działanie licznika, kąt dla którego ma nastąpić dostarczenie paliwa oraz potrzebny czas otwarcia wtryskiwacza. Dla tych danych wzór (12) można przekształcić do postaci:

$$ARR = \frac{\Delta_\alpha}{\alpha_{trig}} m \Delta_{n\_reg} + t_{ON} * f_{TIM} - 1 \quad (15)$$

gdzie:

$\Delta_\alpha$  – różnica między kątami silnika dla rozpoczęcia licznika oraz rozpoczęcia wtrysku,

$\alpha_{trig}$  – kąt silnika odpowiadający jednemu sygnałowi położenia,

$m$  – stosunek częstotliwości zegarów: licznika położenia i licznika wtrysku,

$\Delta_{n\_reg}$  – prędkość silnika podana w różnicy między wartościami rejestrów liczących licznika położenia,

$t_{ON}$  – czas otwarcia wtryskiwacza,

$f_{TIM}$  – częstotliwość taktowania modułu licznika.

Czas wykonywania obliczeń ze wzoru (15) jest dłuży od czasu opisanego dla równania (14) ze względu na dodatkowe człony. Wzór (13) jest zaimplementowany w taki sam sposób, jednak zmienna  $t_{pulse}$  oznacza w tym przypadku czas otwarcia wtryskiwacza.

### 3.2.7 Algorytm Speed Density

Do wyznaczenia obciążenia silnika wybrano algorytm speed density ze względu na fabryczne wyposażenie silnika we wszystkie potrzebne czujniki, a także na jego popularność. Tak jak zostało to opisane w podrozdziale 1.6, algorytm ten wykorzystuje ciśnienie w kolektorze dolotowym oraz panującą tam temperaturę do wyznaczenia masy powietrza, znajdującego się w cylindrze po suwie dolotu. W tym celu korzysta się z równania Clapeyrona [4] przedstawionego we wzorze (16).

$$m = \frac{pVM}{RT} \quad (16)$$

gdzie:

$m$  – masa gazu,

$p$  – ciśnienie gazu,

$V$  – objętość gazu,

$M$  – masa molowa gazu,

$R$  – uniwersalna stała gazowa,

$T$  – temperatura gazu.

Ciśnienie gazu jest wartością odczytaną przez czujnik MAP i określa z jaką siłą powietrze jest wtłaczane do cylindrów. Przy pełnym otwarciu przepustnicy osiąga ona największą wartość (w tym przypadku ciśnienia atmosferycznego). Temperatura jest odczytana przez czujnik IAT, objętość gazu to objętość pojedynczego cylindra, zaś masa molowa przyjęta jest dla powietrza suchego.

Znając masę powietrza znajdującego się w cylindrze, możliwe jest wyliczenie potrzebnej masy paliwa do uzyskania mieszanki stochiometrycznej. Można to wyliczyć ze wzoru:

$$m_{benz} = \frac{m_{pow}}{14,7} \quad (17)$$

gdzie:

$m_{benz}$  – masa benzyny,

$m_{pow}$  – masa powietrza w cylindrze.

Potrzebny czas otwarcia wtryskiwacza w celu dostarczenia danej masy paliwa zależy od jego wydajności. Stała ta wskazuje, jaką objętość paliwa dla danego ciśnienia wejściowego, jest w stanie dostarczyć w przeciągu sekundy. Czas otwarcia może więc być obliczony ze wzoru:

$$t_{wtrysku} = \frac{m_{benz}}{\rho \cdot flow} \quad (18)$$

gdzie:

$m_{benz}$  – masa benzyny,

$\rho$  – gęstość benzyny,

$flow$  – wydajność wtryskiwacza.

Obliczona ze wzoru (18) wartość, nie ma jednak zastosowania w rzeczywistych układach sterowania silnikami spalinowymi. Wynika to przede wszystkim ze zjawiska sprawności napełniania silnika powietrzem (VE) opisanego w rozdziale 1.6. Wartość ta jest charakterystyczna dla danego silnika i jego osprzętu. Zmiana np. charakterystyk wałów rozrządu, kolektora dolotowego czy kształtu komory spalania lub kanałów wlotowych, może znacząco zmienić jego wartość w różnych zakresach pracy silnika. Dodatkowo jak zostało to opisane w rozdziale 1.2, zazwyczaj mieszanka stochiometryczna nie jest pożądana, a jej dokładna wartość zależy od parametrów pracy silnika. Korygowane jest to za pomocą parametru VE. Zależy on od prędkości obrotowej i ciśnienia w kolektorze dolotowym, co pozwala na dokładne opisanie warunków pracy silnika i dlatego jest przedstawiony w kodzie jako trójwymiarowa tabela. Ostateczny wzór zaimplementowany w kodzie został przedstawiony w równaniu (19).

$$t_{wtrysku} = \frac{m_{benz}}{\rho \cdot flow} \cdot VE \cdot m \quad (19)$$

gdzie:

VE – współczynnik sprawności napełniania silnika powietrzem [%],

m – współczynnik korygujący.

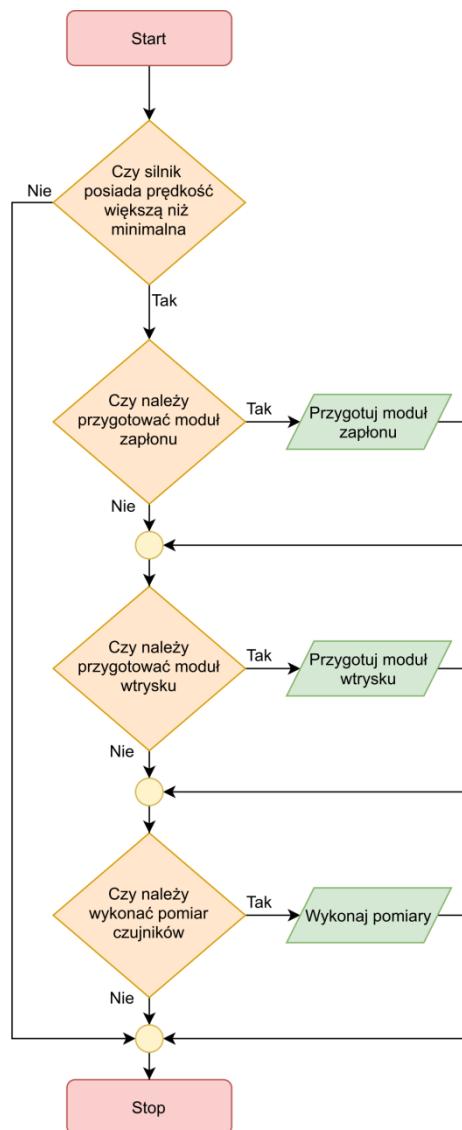
Współczynnik korygujący umożliwia wzbogacenie mieszanki dla szczególnych parametrów pracy silnika. Nominalnie wynosi 1, jednak sumowany jest z wartością odczytaną z tabeli wzbogacenia mieszanki w funkcji temperatury oraz wzbogacenia mieszanki podczas rozruchu silnika. Może być też wykorzystany w trakcie nagłego wciśnięcia pedału gazu, algorytmów ograniczenia prędkości obrotowej, zapobieganiu spalaniu stukowego itp.

### 3.2.8 Logika działania

Działanie kodu oparte zostało o dyskretne momenty czasu, w których pojawiają się przerwania od sygnału prędkości obrotowej silnika. Pozwala to na zachowanie ciągłej synchronizacji z działającym silnikiem oraz wyznaczenie punktów odniesienia dla mechanizmów planujących wykonywanie akcji takich jak zapłon czy podanie paliwa. Dlatego też przerwanie od licznika prędkości obrotowej posiada największy priorytet. Zapewnia to maksymalną precyzję momentu rozpoczęcia działania liczników zapłonu i wtrysku.

Pętla główna programu, oczekuje na pojawienie się zdarzenia za pomocą polecenia asemblerowego WFI (*Wait For Interrupt*). Pozwala ono na zatrzymanie działania potoku, dzięki czemu w momencie wystąpienia przerwania, liczba taktów zegara potrzebna na rozpoczęcie procedury ISR, jest zmniejszona o czas potrzebny na dokończenie wykonywania poprzednich instrukcji (np. NOP). Dodatkowo polecenie WFI zmniejsza zużycie prądu przez rdzeń w czasie, gdy nie jest on wykorzystywany [23].

Po wyjściu mikrokontrolera ze stanu czekania i obsłużeniu przerwania od licznika prędkości obrotowej, wykonywana jest logika związana z rozpoczęciem przygotowania zdarzeń przedstawiona na rys. 32.



Rys. 32 Schemat logiki wykonywanej po obsłudze przerwania od licznika prędkości obrotowej

Jest ona wykonywana tylko dla pewnych prędkości obrotowych silnika, dla których zakłada się rozpoczęcie procesu rozruchu (np. 250[RPM]). Moment, dla którego rozpoczyna się przygotowanie zapłonu lub wtrysku jest ściśle określony. Kąty, dla których to następuje oraz związane z tym charakterystyczne wartości, zostały przedstawione w Tabela 3.

**Tabela 3 Charakterystyczne wartości kątów silnika**

Numer cylindra	1	3	2
Kąt silnika dla GMP suwu kompresji	0°	240°	480°
Kąt silnika w którym przygotowywany jest zapłon	480°	0°	240°
Kąt silnika w którym rozpoczyna się suw dolotu	360°	600°	120°
Kąt silnika w którym kończy się suw dolotu	540°	60°	300°
Kąt silnika w którym przygotowywany jest wtrysk	300°	540°	60°

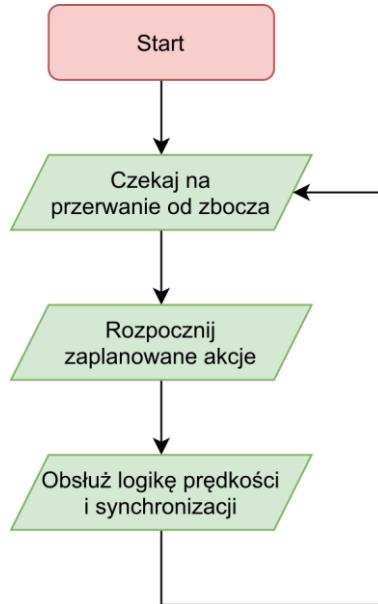
Rozpoczęcie przygotowania danego modułu rozpoczyna się, jak to można zauważyć, w chwili zakończenia działania dla poprzedniego cylindra. Pozwala to na uzyskanie odpowiedniego czasu na przygotowania oraz ustalenie się parametrów pracy silnika. Gdy sterownik wykryje kąt, dla którego musi rozpocząć przygotowania, odczytuje potrzebne dane (np. prędkość obrotową, wartości z czujników) i wykonuje obliczenia zakładając rozpoczęcie działania liczników podczas kolejnego przerwania jak zostało to opisane w podrozdziałach 3.2.5 i 3.2.6. Jeżeli nie przygotowano ani modułu zapłonu ani wtrysku, zlecane jest rozpoczęcie pomiarów czujników.

Tak jak to zostało przedstawione w Tabela 3, kąty GMP suwu kompresji dla kolejnych cylindrów przesunięte są względem siebie o 240°. Tak więc kąt zapłonu wynoszący 10° oznacza, że podanie iskry nastąpi dla pierwszego cylindra przy kącie położenia silnika wynoszącym 710°, dla kolejnego cylindra będzie to wartość 230°, a dla ostatniego 470°.

Schemat blokowy działania procedury obsługi przerwania licznika prędkości obrotowej został przedstawiony na rys. 33. Pierwszą akcją wykonywaną po wykryciu przerwania jest rozpoczęcie działania przygotowanych liczników. Jest to instrukcja

## Oprogramowanie

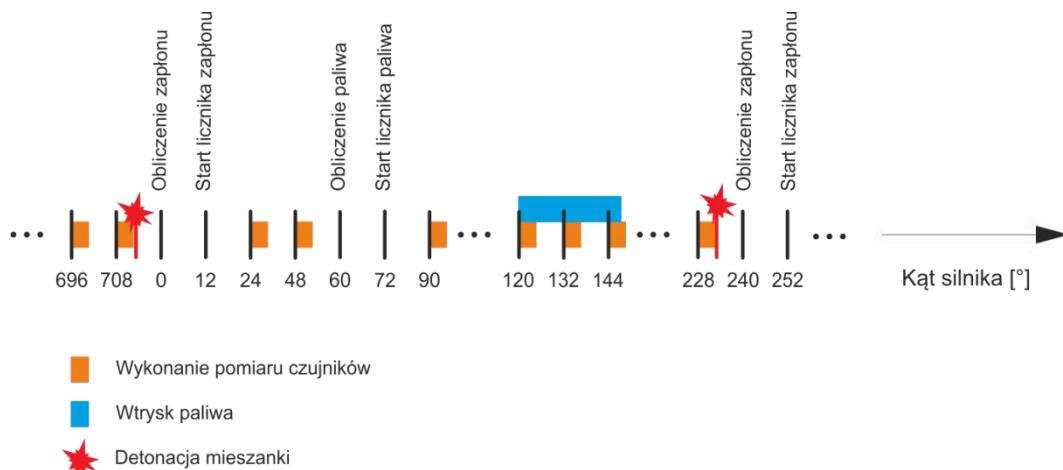
wpisania wartości do jednego rejestru. Tak więc błąd rozpoczęcia działania licznika to kilka cykli zegara, co dla taktowania równego 100[MHz] daje niezauważalną różnicę. Następnie wykonywana jest reszta logiki opisana w podrozdziale 2.3.3.



Rys. 33 Schemat działania obsługi przerwania od licznika prędkości obrotowej

Należy zauważyć, że tylko punkt rozpoczęcia działania mechanizmu, jest wyznaczany przez sygnały położenia. Liczniki mogą wykonać swoje zadanie w dowolnym kącie pracy silnika, z uwzględnieniem rozdzielczości wynikającej z częstotliwości taktowania. Wszystkie opisane mechanizmy pozwalają na uzyskanie wysokiej precyzji.

Przykład działania algorytmu sterowania silnikiem został przedstawiony na rys. 34. Czasy przedstawiono poglądowo i nie odzwierciedlają one rzeczywistych zależności.



Rys. 34 Obrazowy schemat działania algorytmu

## Rozdział 4

### Weryfikacja działania prototypu

Po procesie tworzenia oprogramowania mikrokontrolera rozpoczęto testy, które podzielono na dwa etapy. W pierwszym z nich za pomocą dostępnych narzędzi programistycznych oraz oscyloskopu, sprawdzono poprawność działania projektu w kontrolowanych warunkach. Na tym etapie poprawione zostały drobne błędy w kodzie. Następnie zaś z wykorzystaniem platformy testowej sprawdzono działanie na silniku.

#### 4.1 Testy w środowisku kontrolowanym

Prototyp oraz peryferyjne układy i czujniki połączono ze sobą. Ruch obrotowy silnika symulowany był wkrętarką jak pokazano na rys. 35.

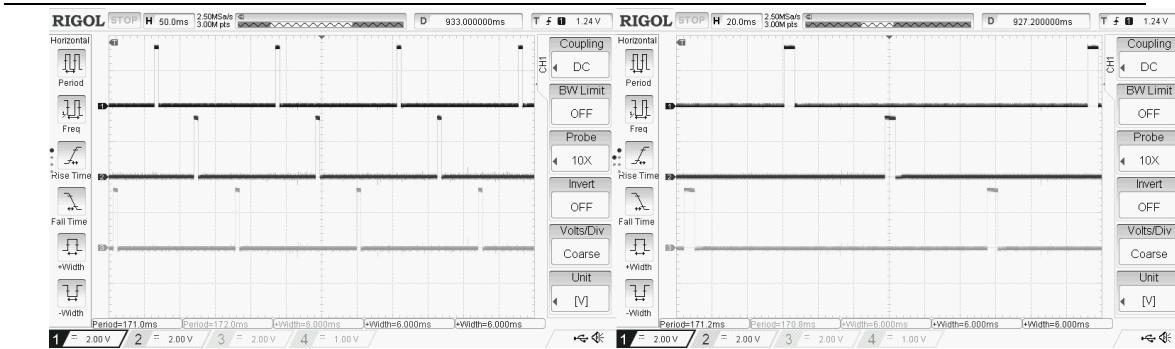


Rys. 35 Test prototypu w warunkach kontrolowanych

Jako pierwsze sprawdzone zostały: moduł tabel, czujników oraz dekodera prędkości i położenia silnika. Za pomocą trybu debugowania porównano znajdujące się w programie wartości z oczekiwanyimi.

W celu sprawdzenia działania modułów sterujących zapłonem, podłączono wyjścia mikrokontrolera do oscyloskopu. Czas ładowania cewek został ustawiony na 6[ms]. Uzyskane w ten sposób oscylogramy przedstawiono na rys. 36.

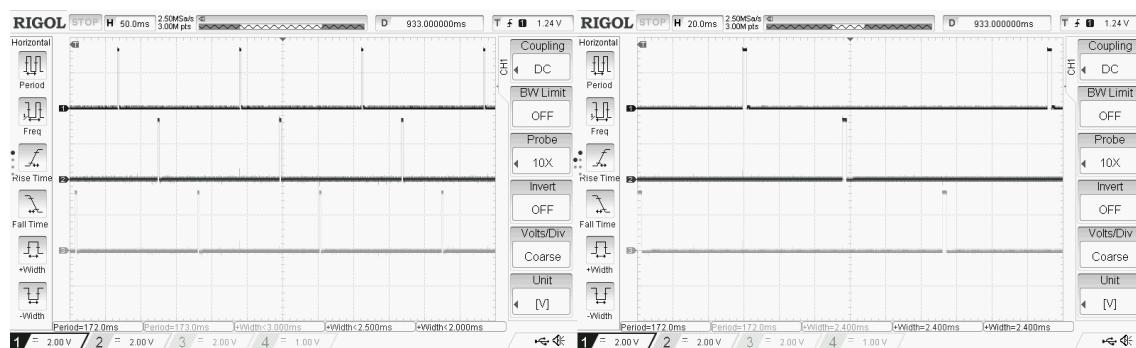
## Weryfikacja działania prototypu



Rys. 36 Oscylogramy przedstawiające sygnały sterowania zapłonem

Przebiegi potwierdzają działanie układu, kanały kolejno załączają się w równych odstępach. Czas ładowania cewek wyznaczony jest bardzo precyzyjnie, oscyloskop odczytuje wartość równą 6.000[ms].

W ten sam sposób przetestowano moduły sterujące działaniem wtrysku paliwa. Uzyskane oscylogramy przedstawiono na rys. 37.



Rys. 37 Oscylogramy przedstawiające sygnały sterowania wtryskiem

Otrzymane czasy wtrysku zostały porównane z ręcznie obliczonymi wartościami dla danych parametrów, co potwierdziło poprawność działania.

Sprawdzono również różnicę w działaniu optymalizacji kodu opisanej w podrozdziale 3.1. Wyniki budowy projektu zostały przedstawione na rys. 38.

text 7856	data 24	bss 1592	dec 9472	hex filename 2508 build/ECU.elf	text 8712	data 24	bss 1592	dec 10328	hex filename 2858 build/ECU.elf
--------------	------------	-------------	-------------	------------------------------------	--------------	------------	-------------	--------------	------------------------------------

Debug

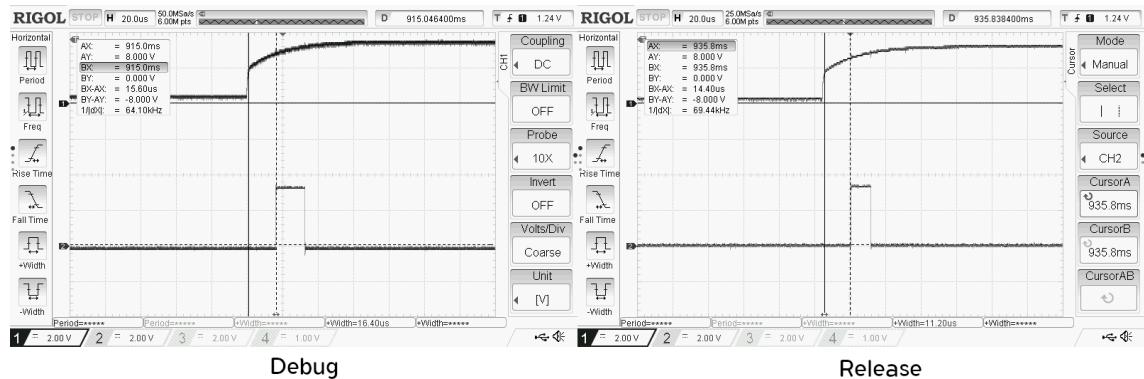
Release

Rys. 38 Porównanie wyniku budowy kodu dla różnych flag optymalizacji

Kod dla celu *Release* jest zdecydowanie większy: posiada więcej instrukcji (*text*). Wynika to z procesu optymalizacji pod kątem szybkości działania np. rozwijanie pętli, usuwanie wywołań funkcji na koszt powtórzenia ich w kilku miejscach w kodzie itd.

## Weryfikacja działania prototypu

Aby sprawdzić prędkość wykonywania kodu dla dwóch przypadków, w momencie rozpoczęcia i zakończenia wykonywania logiki przedstawionej na rys. 32, zmieniany był stan pewnego pinu wyjściowego. Uzyskane oscylogramy przedstawiono na rys. 39. Przebieg czasu wykonywania został odniesiony do sygnału położenia wału korbowego (górnny przebieg).



Rys. 39 Porównanie czasów wykonywania kodu dla różnych flag optymalizacji

Oscylogramy przedstawiają najdłuższą w wykonaniu część logiki, czyli wyliczenie parametrów wtrysku. Markery wskazują czas od wystąpienia przerwania do rozpoczęcia działania logiki. Odczytane wartości zebrane zostały w Tabeli 4.

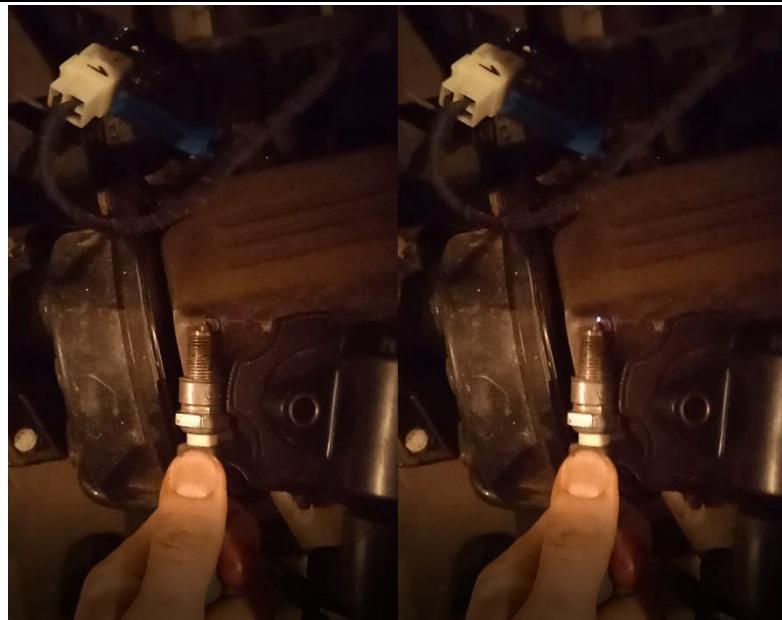
Tabela 4 Porównanie czasów wykonywania kodu dla różnych flag optymalizacji

Parametr	Debug	Release
Czas trwania obsługi przerwania	15,6[ $\mu$ s]	14,4[ $\mu$ s]
Czas wykonywania logiki wyliczenia parametrów wtrysku	16,4[ $\mu$ s]	11,2[ $\mu$ s]

Można zauważyć działanie optymalizacji kodu, dodatkowo zgodnie z założeniami, czasy wykonywania operacji są bardzo krótkie.

## 4.2 Testy na platformie sprzętowej

Kolejnym etapem testów był montaż prototypu na przygotowanym stanowisku testowym. Jako pierwsze sprawdzono czy na elektrodach świec pojawia się łuk elektryczny. W tym celu załączono rozrusznik i obserwowano wykręconą z głowicy świecę przyłożoną do uziemionej pokrywy zaworów. Uzyskaną iskrę pokazano na rys. 40.



**Rys. 40 Test układu zapłonowego**

Następnie w celu sprawdzenia synchronizacji układu, sprawdzono kąt zapłonu z wykorzystaniem lampy stroboskopowej. Podłącza się ją do przewodu zapłonowego pierwszej świecy, w momencie pojawiienia się iskry generuje ona mocny rozbłysk światła. Skierowane jest ono na koło pasowe i obudowę rozrządu, gdzie znajduje się skala z kątami wyprzedzenia zapłonu. W momencie oświetlania widoczne jest w jakim punkcie względem skali znajduje się wskaźnik koła pasowego. Kąt zapłonu dla powtarzalności testu, został zablokowany w kodzie na wartość  $10^\circ$ . Uzyskany efekt został przedstawiony na rys. 41.



**Rys. 41 Test kąta zapłonu**

## Podsumowanie i wnioski

Założone cele zostały zrealizowane. Jak zostało to przedstawione w Rozdział 4 sterownik synchronizuje się z silnikiem, generuje zapłon i włącza na wymagany czas wtryskiwacze. Uzyskano dużą szybkość działania kodu z jednoczesnym zachowaniem wysokiej precyzji działania. Zostało to udowodnione zarówno w warunkach symulacyjnych jak i na przygotowanym w tym celu stanowisku pomiarowym z silnikiem.

Dzięki tej pracy, możliwe było pogłębienie wiedzy z zakresu wykorzystania narzędzi GNU Arm Embedded Toolchain, w szczególności pisania plików *make* i konfigurowania kompilatora GCC. Dodatkowo poznano dokładnie zasady działania mikrokontrolera STM32F411, dzięki pisaniu kodu z wykorzystaniem rejestrów.

Praca może być dalej rozwijana, możliwe jest porównanie różnych algorytmów wyznaczania obciążenia silnika, implementacja różnych funkcji wykorzystywanych w sterownikach silników spalinowych np. wykrywania spalania stukowego co wymaga zastosowania algorytmów DSP. Możliwa jest także implementacja i testowanie algorytmów i układów zabezpieczających przed zakłóceniami elektromagnetycznymi, które są często spotykane w motoryzacji.

## Bibliografia

- [1] W. Jeżewski, *abc samochodowego silnika czterosuwowego*, Warszawa: WKŁ, 1977, str. 9-13
- [2] K. Trzeciak, *Świece zapłonowe*, Warszawa: WKŁ 1989
- [3] S. Radzimirski, *Układy zasilania gaźnikowych silników samochodowych*, Warszawa: WKŁ, 1974
- [4] R. Resnick, D. Halliday, *Fizyka Tom I*, Warszawa: PWN, 1975
- [5] Daihatsu Motor Co., LTD, *Workshop manual CB-23, CB-61 & CB-8*, 1987
- [6] K. Trzeciak, *Obsługa I Naprawa Daewoo Tico*, Warszawa: Wydawnictwo Auto, 1998
- [7] C. Bell, *Maximum Boost: Designing, Testing and Installing Turbocharger Systems*, Cambridge MA: Bentley Publishers, 1997
- [8] Bosch Motorsport, <https://www.bosch-motorsport.com/content/downloads/Raceparts/en-GB/109952523.html>, stan na dzień: 10.12.2021
- [9] R. N. Brady, *Internal Combustion (Gasoline and Diesel) Engines*, Usa: Elsevier Science, 2004
- [10] R. Bosch, *Automotive Handbook*, USA: Wiley, 2019
- [11] SAE International, [https://www.sae.org/standards/content/j1979\\_201408/](https://www.sae.org/standards/content/j1979_201408/) stan na dzień 12.12.2021
- [12] J. Hartman, *How to Tune and Modify Engine Management System*, USA: Motorbooks, 2013
- [13] Daewoo Motor Co., LTD, *Service Manual Matiz*, Korea, 2003
- [14] Pierburg, <https://mam.ms-motorservice.com/mc/epaper?guid=14c101475fa5e02c>, stan na dzień 19.12.2021
- [15] STMicroelectronics, <https://www.st.com/resource/en/datasheet/stm32f411ce.pdf>, stan na dzień 20.12.2021
- [16] WeAct, <https://github.com/WeActTC/MiniSTM32F4x1>, stan na dzień 20.12.2021
- [17] Nissan Motor Co., LTD, *Service Manual Nissan Stanza*, Japan, 1989
- [18] Fairchild, <https://www.onsemi.com/pdf/datasheet/isl9v2540s-f085-d.pdf>, stan na dzień 21.12.2021
- [19] Microsoft, <https://code.visualstudio.com/>, stan na dzień 12.22.2021

- [20] ARM, <https://developer.arm.com/tools-and-software/open-source-software/developer-tools/gnu-toolchain/gnu-rm>, stan na dzień 12.22.2021
- [21] GNU, <https://www.gnu.org/software/make/>, stan na dzień 12.22.2021
- [22] OpenOCD, <https://openocd.org/doc/pdf/openocd.pdf>, stan na dzień 12.22.2021
- [23] J. Yiu, *The Definitive Guide to ARM Cortex–M3 and ARM Cortex–M4 Processors*, Holandia: Elsevier Science, 2014
- [24] CMG Lee,  
[https://commons.wikimedia.org/wiki/File:Comparison\\_of\\_1D\\_and\\_2D\\_interpolation.svg](https://commons.wikimedia.org/wiki/File:Comparison_of_1D_and_2D_interpolation.svg), stan na dzień 26.12.2021
- [25] STMicroelectronics, [https://www.st.com/resource/en/reference\\_manual/rm0383-stm32f411xce-advanced-armbased-32bit-mcus-stmicroelectronics.pdf](https://www.st.com/resource/en/reference_manual/rm0383-stm32f411xce-advanced-armbased-32bit-mcus-stmicroelectronics.pdf), stan na dzień 26.12.2021

## Spis ilustracji

Rys. 1 Charakterystyczne położenie tłoka [1] .....	9
Rys. 2 Schemat działania czterosuwowego silnika o zapłonie iskrowym [1] .....	9
Rys. 3 Przebieg spalania stukowego [2] .....	13
Rys. 4 Schemat gaźnika elementarnego [3].....	14
Rys. 5 Schemat ideowy gaźnika samochodu Daihatsu Charade G11 turbo [5] .....	15
Rys. 6 Schemat ideowy układu zapłonowego samochodu Daewoo Tico [6] .....	16
Rys. 7 Schemat nowoczesnego systemu sterowania silnikiem firmy Bosch [7] .....	17
Rys. 8 Silnik Daewoo F8CV [13].....	20
Rys. 9 Prace przy obudowie tarczy sprzęgłowej .....	21
Rys. 10 Silnik z układem chłodzenia.....	22
Rys. 11 Rama nośna silnika.....	22
Rys. 12 Silnik przymocowany do ramy.....	23
Rys. 13 Układ chłodzenia zamontowany na ramie .....	23
Rys. 14 Układ wydechowy .....	24
Rys. 15 Prototyp sterownika silnika spalinowego .....	25
Rys. 16 Schemat układu kondycjonowania sygnału z czujnika MAP .....	26
Rys. 17 Schemat układu do pomiaru temperatury .....	27
Rys. 18 Przekrój optycznego czujnika położenia silnika w aparacie zapłonowym [17]	28
Rys. 19 Projekt tarczy oraz wykonana tarcza w aparacie zapłonowym .....	29
Rys. 20 Schemat układu wejściowego dla sygnału położenia.....	29
Rys. 21 Oscylogramy przedstawiające sygnały z czujnika położenia wału korbowego	30
Rys. 22 Schemat cewek zapłonowych [13] .....	31
Rys. 23 Symbol tranzystora IGBT ISL9V2540 [18] .....	31
Rys. 24 Schemat wtryskiwacza paliwa [17] .....	32
Rys. 25 Widok konsoli po wywołaniu budowy projektu przez narzędzie make .....	35
Rys. 26 Przykład pliku nagłówkowego stworzonego według szablonu .....	35
Rys. 27 Przykład deklaracji oraz definicji funkcji.....	36
Rys. 28 Ostrzeżenie uzyskane przez podanie zmiennej złego typu dla markera.....	38
Rys. 29 Przykład działania biblioteki SWO .....	38
Rys. 30 Porównanie metod interpolacji: liniowej i biliniowej [24].....	39
Rys. 31 Zasada działania licznika sterującego cewką zapłonową [25] .....	44

## Spis ilustracji

---

Rys. 32 Schemat logiki wykonywanej po obsłudze przerwania od licznika prędkości obrotowej .....	49
Rys. 33 Schemat działania obsługi przerwania od licznika prędkości obrotowej .....	51
Rys. 34 Obrazowy schemat działania algorytmu.....	51
Rys. 35 Test prototypu w warunkach kontrolowanych .....	52
Rys. 36 Oscylogramy przedstawiające sygnały sterowania zapłonem.....	53
Rys. 37 Oscylogramy przedstawiające sygnały sterowania wtryskiem .....	53
Rys. 38 Porównanie wyniku budowy kodu dla różnych flag optymalizacji .....	53
Rys. 39 Porównanie czasów wykonywania kodu dla różnych flag optymalizacji .....	54
Rys. 40 Test układu zapłonowego .....	55
Rys. 41 Test kąta zapłonu .....	55

## Spis tabel

Tabela 1 Wpływ temperatury na graniczne wartości współczynnika $\alpha$ dla benzyny [3]	11
Tabela 2 Parametry zboczy sygnału położenia wału korbowego .....	30
Tabela 3 Charakterystyczne wartości kątów silnika.....	50
Tabela 4 Porównanie czasów wykonywania kodu dla różnych flag optymalizacji .....	54