Электромобильдерге арналған микроконтроллер жүйелерін бағдарламалау

Дәуренов ӘД

4-дәріс. GPIO және электромобиль микроконтроллерлерінің цифрлық кірістері мен шығыстары

Дәрістің мақсаты:

электромобильдің микроконтроллерлері

- сыртқы құрылғылардан цифрлық сигналдарды қалай қабылдайтынын,
- оларды қалай өңдейтінін және
- атқарушы құрылғыларды басқару үшін командаларды қалай қалыптастыратынын түсіндіру.

1. Электромобильдің VCU жүйесіндегі цифрлық кірістер мен шығыстардың рөлі

VCU (Vehicle Control Unit) — электромобильдің әртүрлі электрондық жүйелерінің жұмысын үйлестіретін басқару блогы.

VCU жүргізушінің басқару командаларын, датчиктердің көрсеткіштерін және басқа электрондық блоктардан алынатын ақпаратты өңдейді.

Осы ақпарат негізінде бағдарламалық қамтамасыз ету (ПО) электромобильдің қажетті жұмыс күйін анықтайды.

Мысалы, контроллер келесі ақпаратты қабылдай алады:

- Қозғалыс режимін таңдау қосқышының күйі.
- Тежеу жүйесінен келетін цифрлық сигнал.
- Жүйенің жұмыс істеуіне рұқсат беретін сигналдардың бар-жоғы.

Алынған деректерді талдағаннан кейін VCU тиісті басқару шешімін қабылдайды.

Нақты өнеркәсіптік мысал: NXP S32K376 BMS/VCU

S32K376 платформасы

NXP Semiconductors компаниясы әзірлеген S32K376 BMS and VCU Proof of Concept платформасы BMS пен VCU функцияларын бір электрондық басқару блогына біріктіреді.



Қолданылатын интерфейстер

Бұл инженерлік үлгіде келесі интерфейстер қолданылады:

- GPIO
- ADC
- MSDI
- HSD
- LSD

S32K376 негізгі мүмкіндіктері

GPIO
Цифрлық кіріс/шығыс (датчиктер, реле, индикация)

ADC
Аналогтық сигналдарды цифрлық мәнге түрлендіру (датчиктер: ток, кернеу, температура және т.б.)

MSDI
Автомобильдік байланыс интерфейстері (мысалы, CAN FD)

HSD
Жоғары жылдамдықты интерфейстер (Ethernet, PCIe)

LSD
Төмен жылдамдықты интерфейстер (UART, SPI, I²C, LIN және т.б.)

S32K376 BMS/VCU (бір электрондық басқару блогы)

BMS функциялары

- Батарея кернеуі
- Батарея тогы
- Температура
- Заряд/разряд басқару
- Қауіпсіздік функциялары



VCU функциялары

- Педаль сигналы (APPS)
- Тарту күшін басқару
- Инверторды басқару
- Электрқозғалтқышты басқару
- Автомобиль логикасы

BMS сенсорлары


Батарея элементтері


Ток датчигі


Кернеу


Температура

ADC / GPIO

ADC / GPIO

CAN FD

VCU сенсорлары


APPS (педаль)


Жылдамдық датчигі (дөңгелек)


Руль бұрышы (қосымма)

...


Жоғары вольтты батарея (HV Battery)


Инвертор


Электрқозғалтқыш

S32K376 – бұл сериялық шығарылатын электромобильдің дайын контроллері емес, автомобиль электроникасын әзірлеуге және тексеруге арналған инженерлік платформа.

GPIO дегеніміз не?

GPIO (General-Purpose Input/Output) — микроконтроллердің жалпы мақсаттағы цифрлық кіріс және шығыс желілері.

GPIO микроконтроллерге сыртқы электрондық құрылғылармен өзара әрекеттесуге мүмкіндік береді.

GPIO екі негізгі қызмет атқарады:

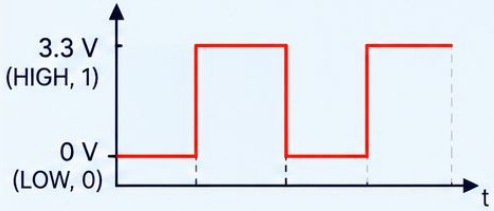
Digital Input — цифрлық кіріс. Сыртқы құрылғыдан келетін логикалық сигналдың күйін анықтайды.

Digital Output — цифрлық шығыс. Микроконтроллер бағдарламасы белгілеген логикалық сигналды сыртқы құрылғыға жібереді.

GPIO: цифрлық сигналдармен жұмыс

Цифрлық сигнал деген не?

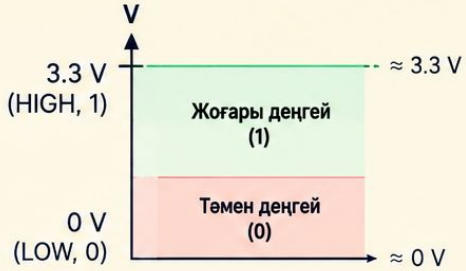
GPIO кірісі/шығысы тек екі логикалық күймен жұмыс істейді: 0 немесе 1.



Логикалық куй деңгейлері

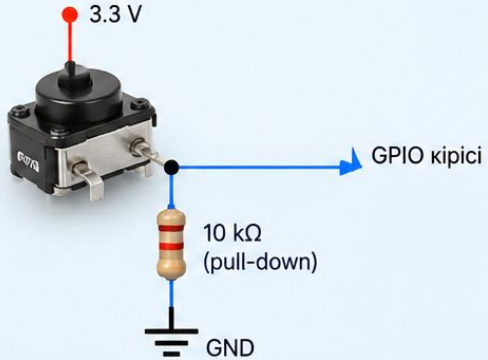
Логикалық күй	Белгіленуі	Түсіндірмесі
0	LOW	Төмен логикалық деңгей
1	HIGH	Жоғары логикалық деңгей

3,3 В жүйесіндегі кернеу деңгейлері

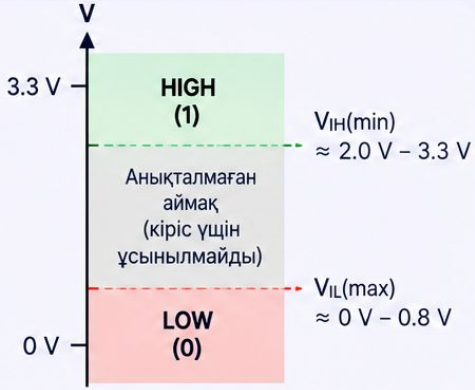


Мысал: түйме қосу (GPIO кірісі)

Түйме басылғанда кезде GPIO кірісінде логикалық деңгей өзгереді.



STM32 GPIO кірісінің жұмыс аймағы (3.3 В)



Маңызды

- 3.3 В логикалық қорек кернеуі бар жүйеде LOW деңгейі шамамен 0 В, ал HIGH деңгейі шамамен 3.3 В болуы мүмкін.
- Нақты логикалық шекті мәндер микроконтроллердің техникалық құжаттамасында (datasheet) көрсетіледі.
- Электромобильдің сыртқы электр тізбектерінде 12 В немесе басқа кернеулер болуы мүмкін.
- Мұндай сигналдарды микроконтроллердің GPIO кірісіне тікелей беруге болмайды.
- Алдымен кернеу деңгейлерін сәйкестендіру және кіріс тізбегін қорғау қажет.

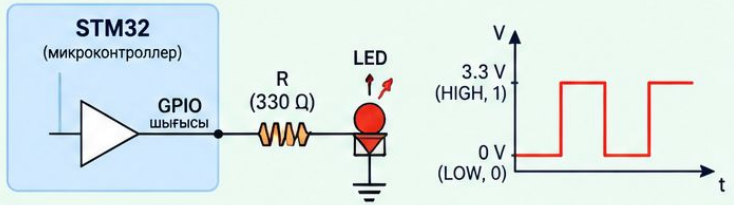
GPIO жұмыс режимдері

Режим	Қызметі	Қолданылуы
INPUT	Цифрлық сигналды қабылдау	Қосқыштың күйін оқу (батырма, датчик)
OUTPUT	Цифрлық сигналды қалыптастыру	Драйверге басқару сигналын жіберу (LED, реле, MOSFET)
INPUT_PULLUP	Кірісті жоғары логикалық деңгейге тарту	GND-ге қосылатын батырма
INPUT_PULLDOWN	Кірісті төмен логикалық деңгейге тарту	Жоғары деңгей беретін қосқыш
OUTPUT_OPEN_DRAIN	Ашық стокты шығыс	Сыртқы тарту резисторы бар үйлесімді интерфейс (I²C және т.б.)
ALTERNATE FUNCTION	Шығысты басқа перифериялық модульге беру	UART, SPI, таймер, CAN және т.б.

INPUT режимі (цифрлық сигналды қабылдау)



OUTPUT режимі (цифрлық сигналды қалыптастыру)

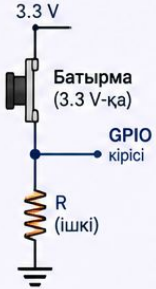


INPUT_PULLUP режимі



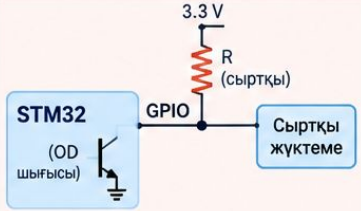
Батырма күйі	GPIO деңгейі
Босатылған (ашық)	HIGH (1)
Басылған (GND-ге)	LOW (0)

INPUT_PULLDOWN режимі



Батырма күйі	GPIO деңгейі
Босатылған (ашық)	LOW (0)
Басылған (3.3 V-қа)	HIGH (1)

OUTPUT_OPEN_DRAIN режимі



- Шығыс LOW деңгейге ғана тарта алады
- Жоғары деңгейге сыртқы резистор арқылы тартылады
- I²C сияқты интерфейстерде қолданылады

ALTERNATE FUNCTION режимі

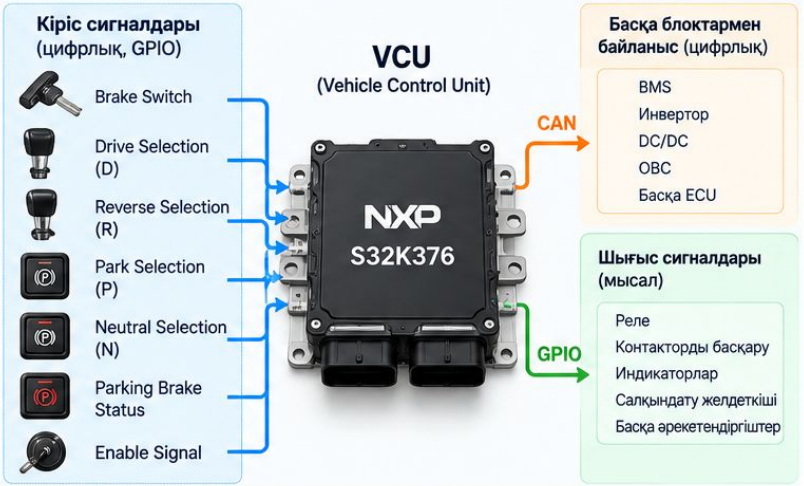


VCU жүйесінде қолданылатын цифрлық сигналдар

Негізгі кіріс сигналдары (цифрлық)

Сигнал	Мақсаты
Brake Switch	Тежеу қосқышының күйін анықтау
Drive Selection	DRIVE режимін таңдау
Reverse Selection	REVERSE режимін таңдау
Park Selection	PARK режимін таңдау
Neutral Selection	NEUTRAL режимін таңдау
Parking Brake Status	Тұрақ тежегішінің күйін анықтау
Enable Signal	Белгілі бір жүйенің жұмыс істеуіне рұқсат беру

Жалпы құрылым (мысал)



Кіріс сигналдарының мысалдары (автомобильде)



Байланыс түрлері

GPIO (цифрлық сигнал)

- Тежегіш, режим таңдау, тұрақ тежегіші, enable сияқты сигналдар әдетте жеке GPIO арқылы беріледі.
- Қарапайым қосқыштар (батырма, датчик) қолданылады.
- Логикалық деңгей: 0 В (LOW) немесе 3.3 В (HIGH).

CAN (немесе басқа шина)

- Кейбір мәліметтер (мысалы, жылдамдық, момент сұранысы, аккумулятор параметрлері) CAN арқылы беріледі.
- Бірнеше ECU арасындағы байланыс үшін қолданылады.
- Нақты архитектура автомобиль өндірушісі мен ECU конфигурациясына байланысты.

Қорытынды: Электромобильдің электрондық басқару жүйесінде көптеген цифрлық сигналдар қолданылады. Олар VCU-ға әдетте жеке GPIO арқылы беріледі, ал кейбір деректер CAN немесе басқа байланыс желілері арқылы беріледі.

MSDI – бірнеше қосқыштың күйін анықтау интерфейсі

Электромобиль электроникасындағы цифрлық қосқыштарды бақылау

1 MSDI не үшін қажет?

Автомобиль электроникасында көптеген цифрлық қосқыштар қолданылады. Олардың әрқайсысын микроконтроллердің жеке GPIO кірісіне тікелей қосу әрдайым тиімді емес.

2 Нақты құрылғы: NXP CD1030

CD1030 – 33 қосқыштың ашық немесе тұйық күйін анықтайтын MSDI микросхемасы. Анықталған күйлерді микроконтроллерге SPI арқылы жібереді.

3 Негізгі идея

MSDI бірнеше сыртқы қосқыштардың күйлерін сенімді түрде жинайды, ал VCU/MCU бұл мәліметтерді өңдеп, басқару логикасында қолданады.



- 33 қосқышты бақылау
- Автомобильдік кернеулерге бейімделген
- Бағдарламаланатын бақылау тогы
- Күй өзгергенде хабарлау
- SPI арқылы байланыс

Дереккөз: NXP CD1030

Жұмыс істеу принципі:

1. Жүргізуші тежегіш педалін басады немесе қозғалыс режимін ауыстырады.
2. Тиісті контактінің электрлік күйі өзгереді.
3. MSDI осы өзгерісті анықтайды.
4. Микросхема контактінің күйі туралы ақпаратты ішкі регистрлерінде сақтайды.
5. MCU ақпаратты SPI интерфейсі арқылы алады.
6. VCU бағдарламасы алынған деректерді әрі қарай тексеру және басқару үшін пайдаланады.

Мысалы: Егер жүргізуші DRIVE (D) режимін таңдаса, VCU тиісті ауыстырып-қосқыштың күйі туралы ақпаратты MSDI арқылы ала алады.

Алайда DRIVE режимін таңдау өздігінен электромобильдің қозғалуына рұқсат берілгенін білдірмейді.

SPI (Serial Peripheral Interface) — микроконтроллер мен сыртқы электрондық құрылғылар арасында деректер алмасуға арналған тізбекті интерфейс.

Біздің жүйеде:

- CD1030 — ауыстырып-қосқыштардың күйін анықтайды.
- SPI — CD1030 мен MCU арасында ақпарат алмасуды қамтамасыз етеді.
- MCU — алынған деректерді өңдейді.

GPIO мен MSDI арасындағы айырмашылық

GPIO	MSDI
MCU-дың әмбебап кірісі/шығысы	Қосқыштарды бақылауға арналған мамандандырылған микросхема
Цифрлық деңгейді тікелей оқи алады	Бірнеше қосқыштың күйін жинайды
Сыртқы сигналға сәйкес қорғаныс қажет	Автомобильдік кіріс тізбектеріне арналған мүмкіндіктері бар
Бағдарлама GPIO күйін оқиды	Бағдарлама SPI арқылы мәлімет алады

Студенттерге арналған маңызды қорытынды: VCU цифрлық ақпаратты тек GPIO арқылы ғана емес, арнайы автомобильдік интерфейс микросхемалары арқылы да ала алады.

MSDI (Multiple Switch Detection Interface) — автомобильдегі бірнеше қосқыштың күйін анықтауға арналған арнайы электрондық интерфейс.

NXP CD1030 микросхемасы 33 қосқышқа дейін бақылай алады және олардың ашық немесе тұйық күйі туралы ақпаратты SPI интерфейсі арқылы микроконтроллерге жібереді.

MSDI автомобильдік кернеулермен жұмыс істеуге, қосқыштарды бақылау тогын реттеуге және күй өзгерген кезде микроконтроллерге хабар беруге мүмкіндік береді.

5. Цифрлық шығыстар және электр жүктемелерін басқару

VCU тек ақпарат қабылдап қана коймай, сыртқы құрылғыларға басқару командаларын да жібереді.

VCU (Микроконтроллер)



GPIO
SPI
PWM

Басқару сигналы (логикалық)

3,3 В / 5 В

Электрондық драйвер

- **High-Side Driver (HSD)**
- **Low-Side Driver (LSD)**

Қуатты қорек (12 В)

Электр жүктемелері

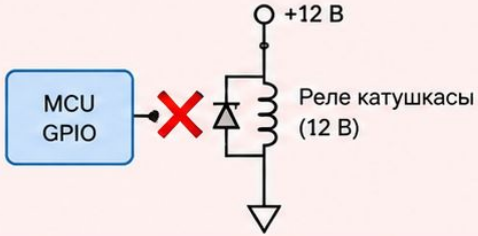

Көмекші реле


Клапан


Сорғы / желдеткіш

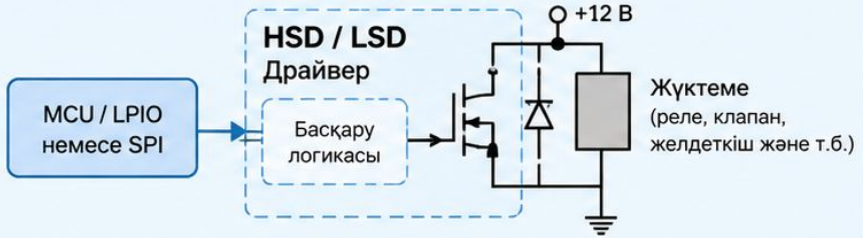

Контактор (жоғары кернеу)

Неліктен GPIO-ға тікелей қосуға болмайды?



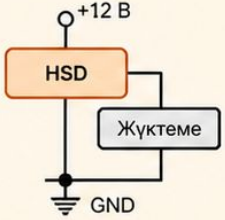
- Ток беру мүмкіндігі шектеулі (әдетте 5–20 мА)
- Кернеу деңгейі сәйкес келмейді (12 В)
- Индуктивті жүктеме кері ЭҚК туғызады
- Микроконтроллер зақымдалуы мүмкін

Драйвер арқылы басқару (дұрыс тәсіл)



High-Side Driver (HSD)

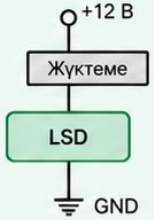
Жүктемені қорек көзінің оң жағынан қосады.



- Жүктемеге +12 В береді
- Қысқа тұйықталудан қорғау
- Диагностика функциялары
- Әдетте SPI арқылы басқарылады

Low-Side Driver (LSD)

Жүктеменің төменгі, яғни қайтару немесе GND жағындағы тізбегін коммутациялайды.



- Жүктемені GND арқылы басқарады
- Бірнеше шығыс арнасы болуы мүмкін
- Қысқа тұйықталудан қорғау
- Диагностика және ақаулықты анықтау
- Әдетте SPI арқылы басқарылады

Нақты автомобильдік мысалдар (NXP)

NXP XS2410
(High-Side Switch)



- Жоғары жақты драйвер
- Автомобильдік қолдану
- SPI арқылы басқару
- Диагностика функциялары

NXP MCZ33996
(Low-Side Switch)



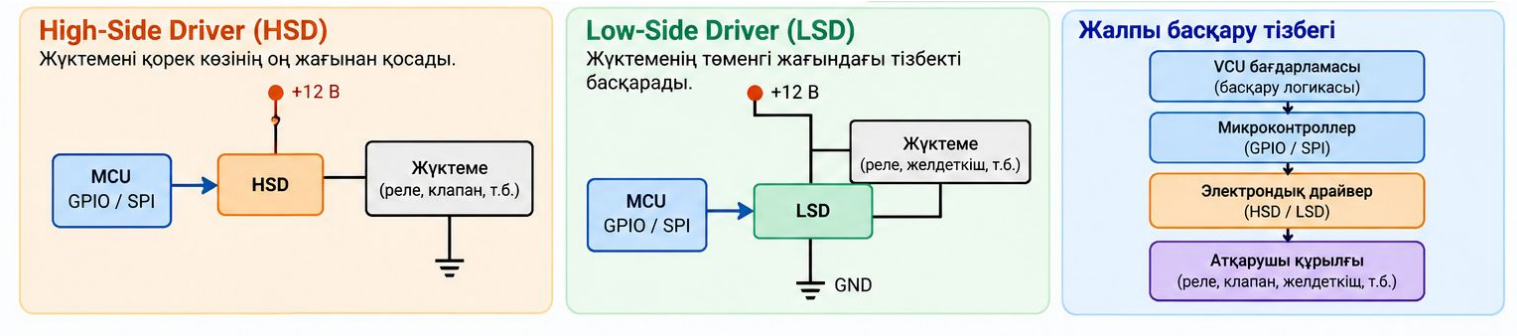
- 16 шығыс арнасы
- Төменгі жақты драйвер
- SPI арқылы басқару
- Ақаулықты анықтау
- Автомобильдік стандарттар

MCU – Басқару командасын қалыптастыру

HSD – Жүктемені қорек көзінің оң жағынан басқару

LSD – Жүктеменің төменгі жағындағы тізбекті басқару

SPI — Драйвермен цифрлық ақпарат алмасу



Электрондық драйвер

Электрондық драйвер (Electronic Driver) — микроконтроллерден (MCU) келетін басқару сигналын қабылдап, электрлік құрылғыны қосуға, өшіруге немесе оның жұмысын басқаруға арналған электрондық құрылғы немесе микросхема.

1. Электрондық драйвер не үшін қажет?

Микроконтроллердің шығысы әдетте аз ток бере алады. Сондықтан ол қуатты электрлік құрылғыларды тікелей басқара алмайды.



Мысалы:
- MCU — 3,3 В басқару сигналын береді.
- Электрондық драйвер — осы сигналға сәйкес сыртқы қуат көзінен жүктемеге ток берілуін басқарады.
- Атқарушы құрылғы — іске қосылады немесе өшіріледі.

Электромобильде қолданылуы

Электрондық драйверлер арқылы MCU келесі құрылғыларды басқара алады:

- Реле және контакторлардың катушкалары.
- Салқындату жүйесінің электрлік клапандары.
- Жарықтандыру құрылғылары.
- Шағын электрқозғалтқыштар мен сорғылар.

CD1030 (MSDI)	Электрондық драйвер
Ауыстырып-қосқыштардың күйін анықтайды	Атқарушы құрылғыны басқарады
MCU-ға ақпарат береді	MCU-дан басқару командасын алады
Кіріс сигналдарымен жұмыс істейді	Шығыс жүктемелерімен жұмыс істейді

Қарапайым түсінік: CD1030 микроконтроллерге не болғанын хабарлайды, ал электрондық драйвер микроконтроллердің командасы бойынша қажетті әрекетті орындайды.

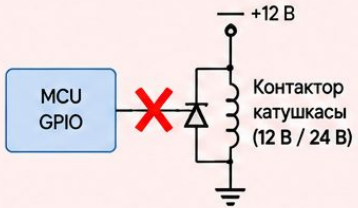
7. Электромобильдің жоғары кернеулі контакторларын басқару

Контакторлар жоғары кернеулі аккумуляторлық тізбектерді қосу және ажырату үшін қолданылады.

Жалпы құрылымдық сұлба



Неліктен контакторды GPIO-ға тікелей қосуға болмайды?



- Катушкадағы ток GPIO мүмкіндігінен (әдетте 5–20 мА) әлдеқайда жоғары (жүздеген мА).
- Индуктивті жүктеме кері ЭҚК түзеді.
- Кернеу деңгейі микроконтроллерге (3,3 В / 5 В) сәйкес келмейді.
- Қорғаныс және диагностика қажет.
- Микроконтроллер зақымдалуы мүмкін.

STMicroelectronics VNH7100BAS драйверінің ішкі құрылымы (жеңілдетілген сұлба)



Контактор катушкасының жұмыс кезеңдері



Жұмыс кезеңдерінің түсіндірмесі

1 Pull-in (іске қосу)
- Контакторды жабу үшін жоғары ток беріледі.
- Әдетте қысқа уақыт (бірнеше ондаған мс).

2 Hold (ұстап тұру)
- Контактор жабылғаннан кейін ток төмендетіледі.
- Энергия тұтынуды азайту үшін ұстап тұру режимі қолданылады.

3 Turn-off (ажырату)
- Катушка тогы өшіріледі.
- Контактор ажыратылады.

4 Қорғаныс және диагностика
- Индуктивті ток қауіпсіз басқарадық.
- Артық ток, қысқа тұйықталу, температура және басқа ақаулықтар тексеріледі.

Нақты өнеркәсіптік мысалдар (NXP)

NXP XS2410
(High-Side Switch)



- Жоғары жақты қосқыш
- SPI арқылы басқару
- Диагностика функциялары
- Автомобильдік қолдану

NXP MCZ33996
(Low-Side Switch)



- 16 шығыс арнасы
- Төменгі жақты қосқыш
- SPI арқылы басқару
- Диагностика және қорғаныс

Қолдану салалары (EV)



Жоғары кернеулі контактор (High-Voltage Contactor) — электромобильдің жоғары кернеулі электр тізбегін қосуға және ажыратуға арналған электромагниттік коммутациялық құрылғы.

Қарапайым тілмен айтқанда, бұл — электромобиль аккумуляторы мен инвертор арасындағы қуатты электрлік қосқыш.

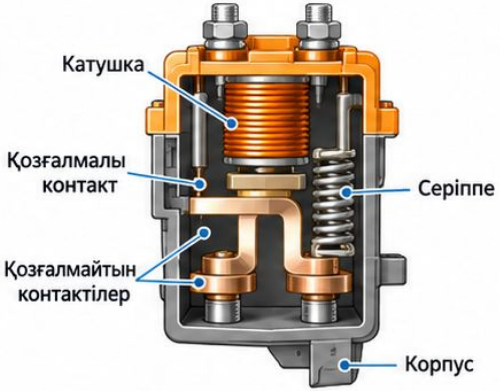
Жоғары кернеулі контактор калай жұмыс істейді?

1. Бұл не?

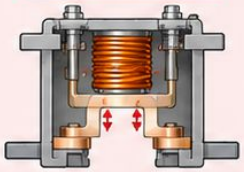


Жоғары кернеулі контактор — электромобильдің жоғары кернеулі тізбегін қосуға және ажыратуға арналған электромагниттік коммутациялық құрылғы.

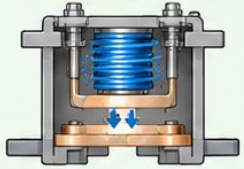
2. Негізгі бөліктері



3. Жұмыс принципі



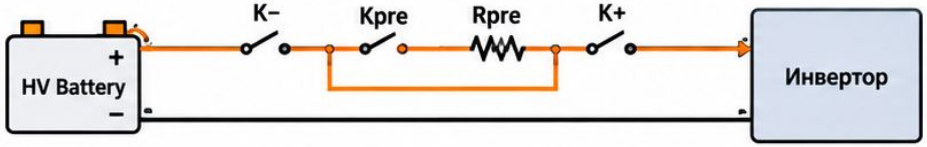
Контактілер ашық
Ток өтпейді



Контактілер жабық
Ток өтеді

1. VCU/BMS рұқсат береді → 2. Драйвер катушкаға 12 В береді
3. Электромагниттік өріс пайда болады → 4. Қозғалмалы контакт тартылады
5. Негізгі тізбек қосылады

4. Электромобильдегі қосылу



Іске қосу реттілігі:

1. K− жабылады
2. Kpre жабылады
3. Конденсаторлар Rpre арқылы зарядталады
4. K+ жабылады
5. Kpre ажыратылады

Алдын ала зарядтау — іске қосу сәтіндегі үлкен токты шектейді.

5. Неге 12 В арқылы 400 В басқарылады?

Бөлік	Кернеу	Қызметі
MCU және драйвер	12 В жүйесі	Катушканы басқарады
Контактор катушкасы	12 В	Контактілерді қозғалтады
Негізгі контактілер	400 В DC	Жоғары кернеулі тізбекті қосады/ажыратады

Қорытынды: MCU → драйвер → катушка → контактор → аккумулятор мен инвертор арасындағы қуат тізбегі қосылады.

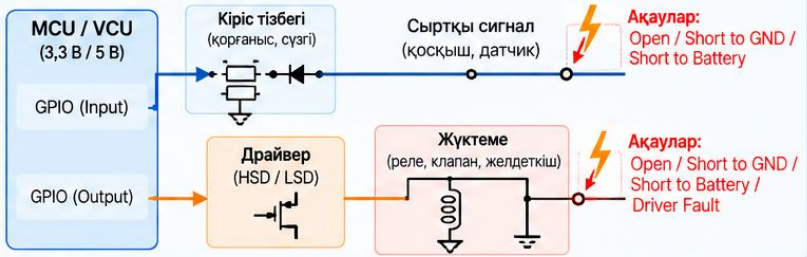
8. Цифрлық кірістер мен шығыстардың диагностикасы

Автомобиль микроконтроллерінің бағдарламасы кірістер мен шығыстарда пайда болуы мүмкін ақауларды анықтауы керек.

Неліктен диагностика қажет?

- Қауіпсіздік пен сенімділікті қамтамасыз ету.
- Ақау кезінде жүйенің дұрыс әрекетін сақтау.
- Драйверлер мен атқарушы құрылғылардың зақымдалуын болдырмау.
- VCU ақау туралы жазба қалыптастырып, тиісті әрекет етеді.

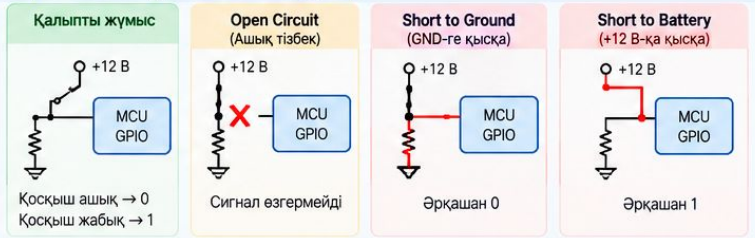
Жалпы қосылу сұлбасы және диагностика аймақтары



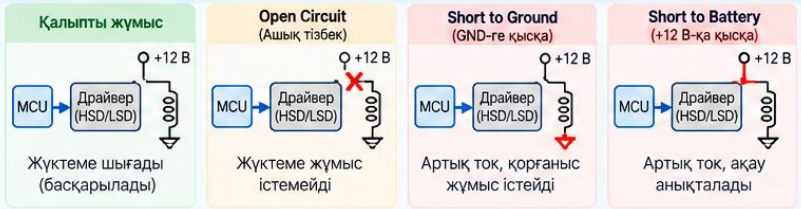
Негізгі ақаулар және олардың белгілері

Ақаулық түрі	Түсіндірме	Мысал
Open Circuit (Ашық тізбек)	Электр тізбегінің үзілуі	Үзілген сым, ажыраған коннектор
Short to Ground (GND-ге қысқа тұйықталу)	Сигнал сымының GND-ге тікелей қосылуы	Сым оқшаулауының бұзылуы
Short to Battery (Қорек көзіне қысқа тұйықталу)	Сигнал сымының +12 В немесе +24 В-қа тікелей қосылуы	Оқшаулаудың зақымдалуы
Invalid State (Рұқсат етілмеген күй)	Күтілмеген немесе тұрақсыз логикалық деңгей	Жоғары шудың әсері, жалған сигнал
Communication Fault (Байланыс ақауы)	Сыртқы микросхемамен байланыстың бұзылуы	SPI, CAN, UART хаттамасы бойынша қате
Driver Fault (Драйвер ақауы)	Атқарушы құрылғы драйверінің ақаулығы	HSD/LSD шығысының қысқа тұйықталуы, токтың артық болуы, температура

Кіріс тізбегіндегі типтік ақаулар



Шығыс тізбегіндегі типтік ақаулар



Уақыт бойынша сигналды тексеру мысалы



9. VCU бағдарламасында цифрлық сигналдарды өңдеу

VCU кіріс сигналдарын қабылдап, олардың күйін тексеріп, қауіпсіз және дұрыс басқару командаларын қалыптастырады.



Цифрлық сигналдарды өңдеудің негізгі кезеңдері

1 Кіріс құрылғыларын бастапқы баптау
- GPIO, SPI, MSDI инициализациясы
- Кіріс параметрлері (Pull-up/Pull-down)
- Дебаунс уақыты

2 Сигналдардың ағымдағы күйін оқу
- 0/1 мәнін оқу
- Күйді периодты түрде жаңарту
- Кіріс фильтрациясы

3 Механикалық контакт дірілін сүзгілеу (дебаунс)
- Жалған ауысуларды жою
- Уақыттық фильтр (мысалы 10–50 мс)
- Тұрақты күйді анықтау

4 Сигналдардың жарамдылығын тексеру
- Open/Short ақауларын анықтау
- Күтілетін диапазонды тексеру
- Бірнеше сигналдың үйлесімділігін тексеру

5 VCU-дың ішкі жұмыс күйін анықтау

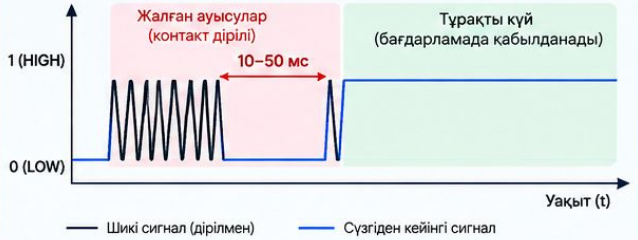
OFF → READY → DRIVE

- Кіріс сигналдары негізінде күйлер
- Жауапкершілік шарттарын тексеру (safety checks)

6 Басқару командаларын қалыптастыру
- Контакторды қосу/ажырату Сорғы, желдеткіш, клапан
- PWM басқару
- Қауіпсіздікке тәуелді шарттар

7 Ақаулықтарды тіркеу және әрекет ету
- DTC (Diagnostic Trouble Code) жазу
- Қорғаныс режиміне өту
- Ақау пайда болғанда қауіпсіз күйді сақтау
- Диагностикалық ақпарат жіберу (CAN)

Контакт дірілін сүзгілеу (дебаунс) мысалы



Ақаулықтарды анықтау мысалдары

Open Circuit (Ашық тізбек)
- Сигнал үнемі 1 (немесе 0)

Short to Ground (GND-ге қысқа)
- Сигнал үнемі 0

Short to Battery (+12 В-қа қысқа)
+12 В
- Сигнал үнемі 1

Invalid State (Рұқсат етілмеген күй)
- Сигналдардың үйлесімсіздігі

NXP Real-Time Drivers (RTD)



- Микроконтроллер перифериясымен жұмыс істеуге арналған драйверлер жиынтығы
- GPIO, SPI, CAN, ADC және басқа модульдерді басқару
- Автомобиль стандарттарына сәйкес әзірленген
- VCU бағдарламасының сенімді жұмысын қамтамасыз етеді

1 Жүргізушінің әрекеті

Жүргізуші тежегішті басады немесе қозғалыс режимін таңдайды (P/R/N/D).



Тежегіш педалі



Қозғалыс режимін таңдау батырмасы

2 Электрлік күй өзгеруі

Тиісті контакт/батырманың электрлік күйі өзгереді (жабық/ашық).



Батырма



Ауыстырып-қосқыш

3 MSDI анықтайды

CD1030 (MSDI) барлық кірістерді бақылайды және күй өзгерісін анықтайды.

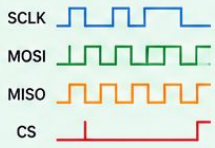


MSDI (мысалы, CD1030)
Ауыстырып-қосқыштардың күйін анықтайды.

SPI

4 SPI арқылы дерек беру

MSDI анықтаған ақпарат SPI интерфейсі арқылы MCU-ға беріледі.



5 MCU өңдейді

STM32 (Blue Pill) алынған деректерді өңдеп, VCU логикасын орындайды.



STM32 Blue Pill (VCU)

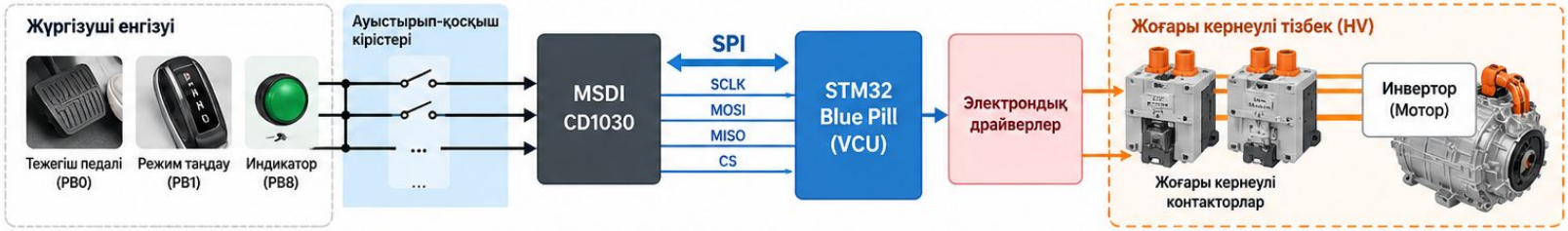
6 Басқару және тексеру

VCU алынған ақпаратты қауіпсіздік тексерулеріне пайдаланып, контакторлар мен басқа жүйелерді басқарады.

- ✔ Тежегіш басылды ма?
- ✔ Режим дұрыс па?
- ✔ Басқа қауіпсіздік шарттары орындалды ма?

Контакторларды басқару (қосу/ажырату)

Жалпы жүйе сұлбасы

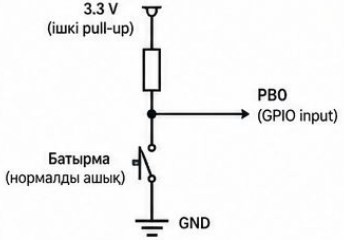


STM32 кіріс сигналдары (оқу мысалы)

STM32 шығысы	Құрылғы
PB0	Тежегіш батырмасы
PB1	DRIVE режимін таңдау батырмасы
PB8	Рұқсат индикаторы

GPIO қосылу сұлбасы (қарапайым мысал)

Батырма → STM32 (ішкі Pull-up қолданылады)



Жоғары кернеулі контактордың құрылысы



Контактордың жұмыс істеуі

Ажыратыллян (OFF)



Катушкаға кернеу жоқ
Контактілер ашық
Ток өтпейді

Қосылған (ON)



Катушкаға 12 В беріледі
Контактілер жабық
Жоғары кернеулі ток өтеді

STM32 көмегімен цифрлық сигналдарды өңдеу

STM32 Blue Pill микроконтроллері арқылы екі батырманың күйін оқу, логикалық шартты тексеру және нәтижеге байланысты жарықдиодты басқару.

Бұл мысалда электромобильдің қозғалыс режимін таңдаудың қарапайым оқу моделі қарастырылады.

```
const int brakePin = PB0;  // PB0 — тежегіш батырмасы қосылған кіріс

const int drivePin = PB1;  // PB1 — DRIVE режимін таңдау батырмасы қосылған кіріс

const int driveIndicator = PB8;  // PB8 — рұқсат индикаторы (LED) қосылған шығыс

void setup() {              // setup() — бағдарлама іске қосылғанда бір рет орындалады

  Serial.begin(115200);       // Serial Monitor арқылы деректерді шығару
                              // 115200 — ақпарат алмасу жылдамдығы (бит/с)

  pinMode(brakePin, INPUT_PULLUP);       // PB0 кіріс ретінде орнатылады
                                         // Ішкі Pull-up резисторы қосылады

  pinMode(drivePin, INPUT_PULLUP);            // PB1 кіріс ретінде орнатылады
                                              // Ішкі Pull-up резисторы қосылады

  pinMode(driveIndicator, OUTPUT);              // PB8 шығыс ретінде орнатылады
                                                // Бұл пин жарықдиодты басқарады
}
void loop() {                                // loop() — бағдарлама жұмыс істеп тұрғанша
                                             // үздіксіз қайталанатын функция

  bool brake = digitalRead(brakePin) == LOW;                 // PB0 пинінің күйін оқиды
                                                             // LOW болса — батырма басылған (true)
                                                             // HIGH болса — батырма басылмаған (false)

  bool drive = digitalRead(drivePin) == LOW;
                                                                          // PB1 пинінің күйін оқиды
                                                                          // LOW болса — DRIVE таңдалған (true)
                                                                          // HIGH болса — DRIVE таңдалмаған (false)

bool requestAccepted = brake && drive;
                                // && — логикалық ЖӘНЕ (AND)
операторы
                                // Екі шарт та true болса, нәтиже true болады
                                // Басқа жағдайларда нәтиже false болады

 digitalWrite(driveIndicator,
          requestAccepted ? HIGH : LOW);

                                        // PB8 шығысын басқарады
                                  // Егер requestAccepted = true болса → HIGH
                              // Егер requestAccepted = false болса → LOW
                                        // ? : — шартты таңдау операторы

  Serial.print("Brake: ");
  // Serial Monitor экранына Brake мәтінін шығарады
  Serial.print(brake);
  // Тежегіштің күйін шығарады: 1 немесе 0
  Serial.print("  Drive: ");
  // Drive мәтінін сол жолға шығарады
  Serial.print(drive);
  // DRIVE батырмасының күйін шығарады: 1 немесе 0
  Serial.print("  Request accepted: ");
  // Request accepted мәтінін шығарады
  Serial.println(requestAccepted);
  // Логикалық тексерудің нәтижесін шығарады
  // println келесі жолға көшеді
  delay(100);
  // Бағдарлама 100 миллисекунд күтеді
  // Содан кейін loop() қайта орындалады
}
```

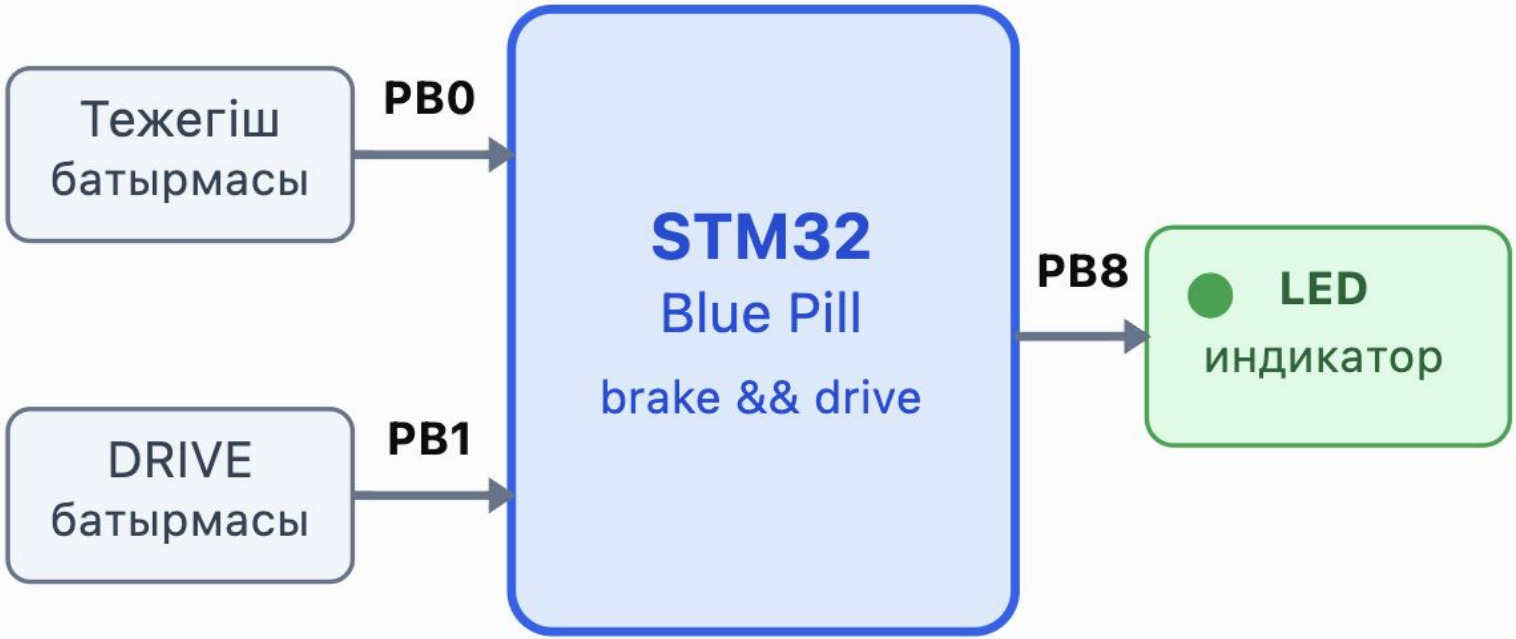
Негізгі командалардың мағынасы

Команда Қызметі

const int – Өзгермейтін бүтін сандық тұрақтыны анықтайды
void setup() – Бастапқы баптауларды орындайды
void loop() – Командаларды үздіксіз қайталайды
pinMode() – Пиннің жұмыс режимін анықтайды
INPUT_PULLUP – Кіріс және ішкі тартушы резистор режимі
digitalRead() – Цифрлық кірістің күйін оқиды
bool – true немесе false мәнін сақтайды
== Екі мәннің теңдігін салыстырады
&& – Логикалық ЖӘНЕ (AND)
digitalWrite() – Цифрлық шығысқа HIGH немесе LOW орнатады
? : Шартқа байланысты екі мәннің бірін таңдайды

Serial.print() — Мәтінді сол жолға шығарады
Serial.println() — Мәліметті шығарып, жаңа жолға көшеді
delay(100) — 100 миллисекунд кідіріс жасайды

STM32 Blue Pill — оқу моделінің логикалық сұлбасы



Екі кіріс → логикалық AND → бір шығыс

Жұмыс реті:

1. STM32 іске қосылып, PB0 және PB1 пиндерін кіріс, PB8 пинін шығыс ретінде баптайды.
2. Микроконтроллер тежегіш пен DRIVE батырмаларының күйін оқиды.
3. Егер батырма басылса, INPUT_PULLUP режимінде оның кірісіндегі сигнал LOW болады.
4. Бағдарлама brake && drive шартын тексереді.
5. Екі батырма да басылған болса, requestAccepted = true болады.
6. STM32 PB8 шығысына HIGH сигналын беріп, индикаторды жағады.
7. Кемінде бір батырма басылмаса, индикатор сөнеді.
8. Бағдарлама 100 мс күтіп, тексеруді қайта орындайды.

Нақты электромобильмен байланысы

Бұл оқу мысалында STM32 батырмалардан сигналдарды тікелей GPIO арқылы оқиды. Ал нақты электромобильде тежегіш пен қозғалыс режимінің сигналдары арнайы кіріс микросхемалары (мысалы, MSDI), басқа ECU немесе байланыс интерфейстері арқылы VCU-ға берілуі мүмкін.

Маңызды: requestAccepted = brake && drive — тек оқу мақсатындағы шарт. Нақты электромобильде қозғалысқа рұқсат беру үшін BMS күйі, жоғары кернеулі жүйенің дайындығы, ақаулардың болмауы және басқа да қауіпсіздік талаптары тексеріледі. Индикатордың жануы нақты электромобильдің қозғалысқа дайын екенін білдірмейді.

Сондай-ақ мысалдағы delay(100) оқу үшін жарамды, бірақ нақты автомобиль контроллерінде уақытқа тәуелді тапсырмалар әдетте таймерлермен және кезеңдік орындалатын бағдарламалық механизмдермен ұйымдастырылады.

Бақылау сұрақтары

1. GPIO дегеніміз не?
2. GPIO-ның INPUT және OUTPUT режимдерінің айырмашылығы қандай?
3. Pull-up және Pull-down резисторлары не үшін қажет?
4. VCU қандай цифрлық сигналдарды қабылдай алады?
5. MSDI интерфейсі не үшін қолданылады?
6. GPIO мен CD1030 микросхемасының айырмашылығы қандай?
7. High-Side Driver және Low-Side Driver қандай қызмет атқарады?
8. Неліктен контактор катушкасын микроконтроллерге тікелей қосуға болмайды?
9. Автомобильдік цифрлық сигналдарды диагностикалау не үшін маңызды?
10. MCU сыртқы цифрлық сигнал негізінде басқару шешімін қалай қабылдайды?