Электромобильдерге арналған микроконтроллер жүйелерін бағдарламалау

Дәуренов ӘД

5-дәріс. ADC және электромобиль датчиктерінің сигналдарын өңдеу

Дәрістің мақсаты:

электромобильдің электрондық жүйелерінде

- аналогтық сигналдарды өлшеу принциптерін,
- аналогтық-цифрлық түрлендіргіштің жұмысын
- және акселератор педалі, температура,
- кернеу мен ток датчиктерінің мәліметтерін бағдарламалық өңдеу әдістерін меңгеру.

1. Электромобильде ADC не үшін қажет?

Электромобильдің электрондық басқару жүйесі көптеген физикалық шамаларды үздіксіз бақылайды.

Алдыңғы дәрісте қарастырылған цифрлық қосқыштардан айырмашылығы, мұндай физикалық шамалардың мәндері үздіксіз өзгеруі мүмкін.

Мысалы:

- Акселератор педалінің басылу дәрежесі 0–100% аралығында өзгеруі мүмкін.
- Аккумулятордың температурасы жұмыс барысында өзгереді.
- Аккумулятор тогы қозғалыс немесе зарядтау режиміне байланысты өзгереді.
- Аккумулятор ұяшықтарының кернеуі зарядталу деңгейіне және жүктемеге байланысты өзгереді.

Бұл шамаларды микроконтроллердің бағдарламасы өңдеуі үшін олардың аналогтық электр сигналдары цифрлық мәндерге түрлендірілуі қажет.

ADC (Analog-to-Digital Converter) — аналогтық-цифрлық түрлендіргіш (АЦТ).

ADC аналогтық электр сигналын цифрлық санға түрлендіреді.

Электромобильдің нақты жүйелерінде ADC әртүрлі жерде орналасуы мүмкін:

1. VCU микроконтроллерінің ішінде.
2. BMS жүйесінің мамандандырылған өлшеу микросхемасында.
3. Датчиктің немесе басқа электрондық құрылғының ішінде.

Сондықтан аналогтық сигнал әрдайым негізгі микроконтроллердің ADC кірісіне тікелей қосыла бермейді.

2. NXP S32K376 платформасындағы ADC: аналогтық кірістерді өңдеу

Акселератор педалінің күші және басқа датчиктердің аналогтық сигналдары микроконтроллердің ADC модуліне жеткізіліп, сандық кодқа айналдырылады.

Жалпы құрылым (ақпараттың ағымы)



Акселератор педалі (APPS)

Аналогтық сигнал (0.5 – 4.5 В)

Кіріс тізбегі (қорғаныс, фильтр, шоғырландыру)

NXP S32K376 (VCU)

- ADC модулі
- Цифрлық мән (мысалы, 12 бит)
- VCU бағдарламасы (сигналды өңдеу, тексеру, диагностика)

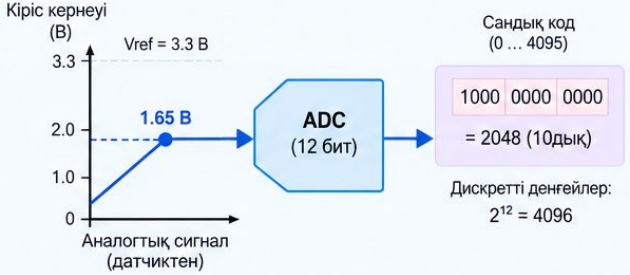
Педаль күйі (0…100 %)

Қауіпсіздік және ақаулықтар

1 Аналогтық сигналдың мысалы (акселератор педалі)



2 ADC: аналогтық сигналды сандық кодқа айналдыру



3 Бағдарламалық өңдеу

1. ADC мәнін оқу
2. Шуды сүзу (фильтрация, орташа мән)
3. Физикалық шамаға айналдыру (мысалы, педаль %)
4. Шектеулер мен қауіпті жағдайларды тексеру (диагностика)
5. VCU логикасына жіберу

4 Нақты мысал: педаль күйін есептеу

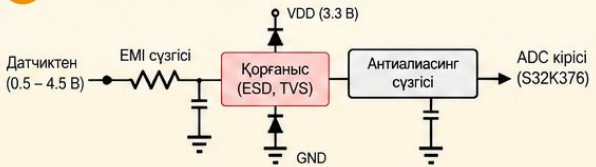
ADC мәні	Кернеу (В) (3.3 В, 12 бит)	Педаль күйі (%)
0	0.00	0
256	0.21	≈ 5
1024	0.83	≈ 20
2048	1.65	≈ 40
3072	2.48	≈ 70
4095	3.30	≈ 100

$$\text{Кернеу} = \text{ADC_мәні} \times \frac{3.3\ \text{В} / 4095}{4095}$$

Неліктен өңдеу қажет?

- Шу мен тербелістерді еңқарту
- Дұрыс педаль күйін есептеу
- Ақаулықтарды анықтау (open/short, қисықтық)
- Қауіпсіздік талаптарын орындау
- Басқару командаларын дұрыс қалыптастыру

5 Кіріс тізбегінің мысалы



6 Кіріс тізбегіндегі ықтимал ақаулықтар

Open Circuit (Ашық тізбек)	Short to GND (GND-ге қысқа)	Short to Battery (+12 В-қа қысқа)	Invalid State (Рұқсат етілмеген күй)
Сигнал үзіледі (өте төмен немесе өте жоғары мән)	Кіріс ≈ 0 В (ADC ≈ 0)	+12 В Кіріс ≈ 3.3 В (ADC ≈ 4095) (қорғаныс тізбегі шектейді)	Алынған сигнал физикалық заңдылыққа сәйкес емес (мысалы, бірден секіру)

7 Қолданылатын драйверлер / RTD

- S32K3 RTD (Real-Time Drivers)
- ADC драйверлері
- GPIO / SIUL2
- Аналогтық сүзгілеу параметрлері
- Қауіпсіздік және диагностика кітапханалары
- Автомобильдік стандарттарға сәйкес жұмыс

Бұл жерде екі маңызды кезеңді ажырату қажет.

Бірінші кезең: ADC аналогтық сигналды цифрлық кодқа айналдырады.

Екінші кезең: VCU бағдарламасы цифрлық кодты физикалық шамаға айналдырып, оның дұрыстығын тексереді.

ADC разрядтылығы — Resolution

ADC-ның негізгі сипаттамаларының бірі — разрядтылық.

Разрядтылық аналогтық сигналдың қанша цифрлық деңгейге бөлінетінін анықтайды.

Формула:

$$N = 2^n$$

Мұндағы:

- N — цифрлық деңгейлер саны;
- n — ADC разрядтылығы.

Разрядтылық	Деңгейлер саны	Цифрлық мәндер
8 бит	256	0–255
10 бит	1024	0–1023
12 бит	4096	0–4095
16 бит	65 536	0–65 535

Мысалы, электромобильде акселератор педалінің орналасу датчигі (APPS — Accelerator Pedal Position Sensor) орнатылған.

Датчик педальдің орналасуына байланысты әртүрлі аналогтық кернеу қалыптастыра алады.

Оқу мысалында:

- Педаль босатылған → 0,5 В
- Педаль жартылай басылған → 1,65 В
- Педаль көбірек басылған → 2,8 В

Микроконтроллер осы кернеуді бағдарлама өңдей алатын цифрлық мәнге түрлендіруі қажет.

Аналогтық кернеу цифрлық мәнге қалай түрленеді?

Мысалы, келесі параметрлер берілсін:

- ADC разрядтылығы — 12 бит.
- Тірек кернеуі — 3,3 В.
- Кіріс кернеуі — 1,65 В.

$$ADC \approx \frac{U_{in}}{3.3} \times 4095$$

Оқу есептеулерінде қолда

$$ADC \approx \frac{1.65}{3.3} \times 4095$$

Мәндерді формулаға қоям

$$ADC \approx 2048$$

Демек, кіріс кернеуі шамамен 1,65 В болғанда, микроконтроллер ADC арқылы шамамен 2048 цифрлық мәнін алады.

ADC электромобильде қалай қолданылады?

Электромобильдегі акселератор педалінің орналасу датчигі (APPS) 0–3,3 В диапазонында сигнал калыптастыратын оқу моделін карастырайық

Кернеу	ADC (12 бит)	Педальдің шартты орналасуы
0 В	0	0%
0,825 В	1024	25%
1,65 В	2048	50%
2,475 В	3071	75%
3,3 В	4095	100%

Бұл — жеңілдетілген оқу моделі.
Нақты электромобильдерде APPS датчиктері көбінесе нөлден өзгеше бастапқы кернеуді қолданады және қауіпсіздік үшін екі бөлек сигнал арнасы болуы мүмкін.

Sampling — аналогтық сигналды дискреттеу

ADC аналогтық сигналды үздіксіз емес, белгілі бір уақыт аралықтарында өлшейді.

Әрбір өлшеу нәтижесі Sample, яғни сигналдың жеке өлшемі деп аталады.

Мысалы, бағдарлама акселератор педалінің күйін әрбір 10 мс сайын оқуы мүмкін.

10 мс — түсіндіру үшін алынған шартты оқу периоды.

Өлшеулер арасындағы уақыт азайған сайын сигналдың өзгерісін жиірек бақылауға болады.

Алайда өлшеу жиілігін таңдағанда келесі факторлар ескеріледі:

- Сигналдың өзгеру жылдамдығы.
- ADC түрлендіру уақыты.
- Өлшеу дәлдігі.
- Электрлік кедергілер.
- Бағдарламалық өңдеу уақыты.
- Жүйенің реакция уақыты.

Электромобильде әртүрлі сигналдардың өлшеу жиіліктері бірдей болмауы мүмкін.

Мысалы, аккумулятор температурасының өзгеруі мен токтың жылдам ауытқуы әртүрлі уақыттық сипаттамаларға ие.

Scaling — аналогтық сигналды масштабтау

ADC сандық мән береді, бірақ VCU бағдарламасына педальдың орналасуын пайызбен анықтау қажет болуы мүмкін.

Ол үшін масштабтау қолданылады.

Егер датчиктің сипаттамасы сызықтық болса:

$$P = \frac{U - U_{min}}{U_{max} - U_{min}} \times 100\%$$

Мұндағы:

- P — педальдың басылу дәрежесі;
- U — өлшенген кернеу;
- U_{min} — педальдың бастапқы күйіне сәйкес кернеу;
- U_{max} — педаль толық басылған кездегі кернеу.

Шартты датчиктің диапазоны:

- Минималды кернеу: 0,5 В.
- Максималды кернеу: 2,8 В.
- Өлшенген кернеу: 1,65 В.

Есептеу:

$$P = \frac{1.65 - 0.5}{2.8 - 0.5} \times 100\%$$

$$P = 50\%$$

APPS (Accelerator Pedal Position Sensor) – акселератор педалінің орналасу датчигі

Жүргізушінің педальды қаншалықты басқанын өлшейді және бұл ақпаратты VCU-ға жібереді.

1 Жалпы жұмыс принципі



Акселератор педалі

APPS (екі арналы датчик)

APP1 (0.5 – 4.5 В)

APP2 (0.5 – 4.5 В)

NXP S32K376 (VCU)

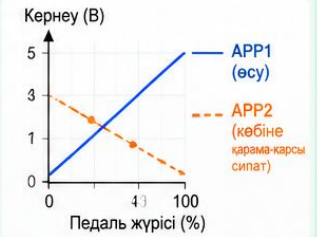
ADC модулі (12/16 бит)

Бағдарламалық өңдеу

Педаль орны (%)

2 APPS ішкі құрылысы (екі арналы)

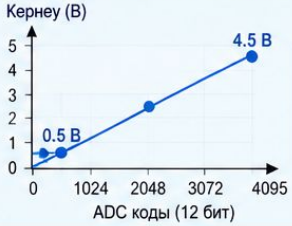




3 APPS сигналдарының сипаттамасы

Параметр	APP1	APP2
Кернеу диапазоны	0.5 – 4.5 В	0.5 – 4.5 В
Сипаттамасы	Сызықты өсу	Сызықты кему (немесе басқа сипат)
Бос күй (0%)	≈ 0.5 – 0.7 В	≈ 4.3 – 4.5 В
Толық басу (100%)	≈ 4.3 – 4.5 В	≈ 0.5 – 0.7 В

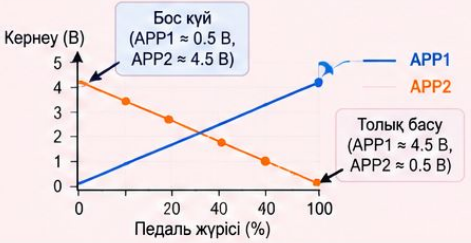
4 Сигналдарды ADC арқылы сандық кодқа айналдыру



Мысалы (12 бит)

- 0.5 В → ≈ 410
- 2.5 В → ≈ 2048
- 4.5 В → ≈ 3686

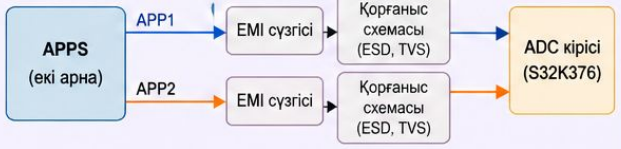
5 Екі арнаның сигналдарын салыстыру



6 VCU бағдарламасында өңдеу кезеңдері

1. ADC мәнін оқу (APP1 және APP2)
2. Фильтрация (шуды сүзу)
3. Педаль орнын есептеу (%)
4. Екі арнаны салыстыру (сенімділік)
4. Қауіпріздік салыстыру (ақауларды анықтау)
5. Қауіпсіздік тексерулері (ақауларды анықтау)
6. VCU басқару логикасы (тарту күші, режимдер)

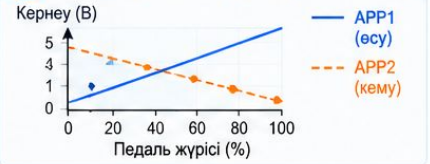
7 Аппараттық қосу сұлбасы (жеңілдетілген)



8 Қауіпсіздік және диагностикалық тексерулер

- APP1 және APP2 диапазондарының дұрыстығы.
- Екі арнаның өзара сәйкестігі (корреляция).
- Сигналдың үзіліуі (open circuit).
- Қысқа тұйықталу (short to GND / to Battery).
- Сигналдың қисығы және жылдам өзгеруі.
- Сенімсіз мәндер кезінде қауіпсіздік режиміне өту.

9 Нақты тәуелділік (дерекқұжат бойынша)



7. Plausibility Checking – сигналдардың сәйкестігін тексеру

VCU екі арнаның сигналдарын салыстырып, олардың дұрыстығын, диапазонын және бір-біріне сәйкестігін тексереді.

1 APPS екі арналы құрылымы



Акселератор педалі (APPS)

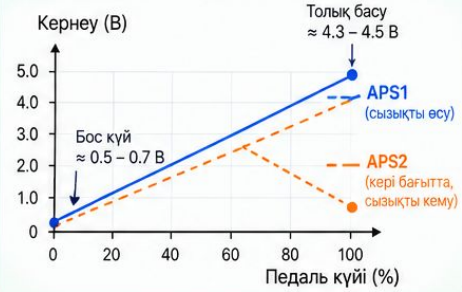
APS1 → Аналогтық сигнал (0.5 – 4.5 В) → ADC модулі (12/16 бит)

APS2 → Аналогтық сигнал (0.5 – 4.5 В) → Plausibility Checking (сәйкестікті тексеру)

NXP S32K376 (VCU)

Plausibility Checking (сәйкестікті тексеру) → Тарту күшін басқару (педаль орны)

2 Қалыпты сигналдардың сипаттамасы



3 Сигналдардың электрлік диапазоны

Параметр	APS1	APS2
Кернеу диапазоны	0.5 – 4.5 В	0.5 – 4.5 В
Бос күй (0%)	≈ 0.5 – 0.7 В	≈ 4.3 – 4.5 В
Толық басу (100%)	≈ 4.3 – 4.5 В	≈ 0.5 – 0.7 В
Сипаттамасы	Сызықты өсу	Сызықты кему

4 VCU-дың тексеру қадамдары

1. Бірінші арнаның сигналын өлшейді.
2. Екінші арнаның сигналын өлшейді.
3. Әр арнаның электрлік диапазонын тексереді.
4. Сигналдарды салыстыруға жарамды өлшем бірлігіне түрлендіреді (пайыз).
5. Екі арнаның өзара сәйкестігін тексереді.
6. Қажет болса, ақаулық күйін анықтайды.

5 Сәйкестік тексеру (шартты оқу мысалы)

Бірінші арна (%)	Екінші арна (%)	Айырмашылық (%)	Нәтиже
30	32	2	✓ Сәйкес
50	51	1	✓ Сәйкес
70	95	25	✗ Сәйкес емес
0	60	60	✗ Сәйкес емес

Бұл мысалда сәйкестікті бағалау үшін 8% пункттік шек қолданылады.

| APS1 – APS2 | ≤ 8% → Сәйкес

| APS1 – APS2 | > 8% → Сәйкес емес (ақаулық)

6 Ақаулық жағдайлары

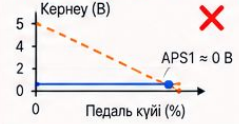
Қалыпты жұмыс –



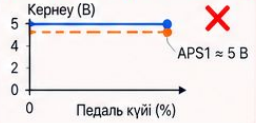
Сигнал үзілуі (Open Circuit)



Қысқа тұйықталу (Short to GND)



Қысқа тұйықталу (Short to Battery)



7 Пайыздық мәнге түрлендіру (нормализация)

$$\text{Педаль күйі (\%)} = \frac{(V_{adc} - V_{min})}{(V_{max} - V_{min})} \times 100$$

$V_{min} \approx 0.5$ В

$V_{max} \approx 4.5$ В

Нәтиже: 0 – 100 %

8 Бағдарламадағы әрекет

APS1 және APS2 оқу (ADC) → Пайызға түрлендіру → Сәйкестікті тексеру (|Δ| ≤ 8%)

- OK → Тарту күшін есептеу (педаль орны)
- Ақаулық → Қауіпсіздік режимі (DTC, шектеулер)

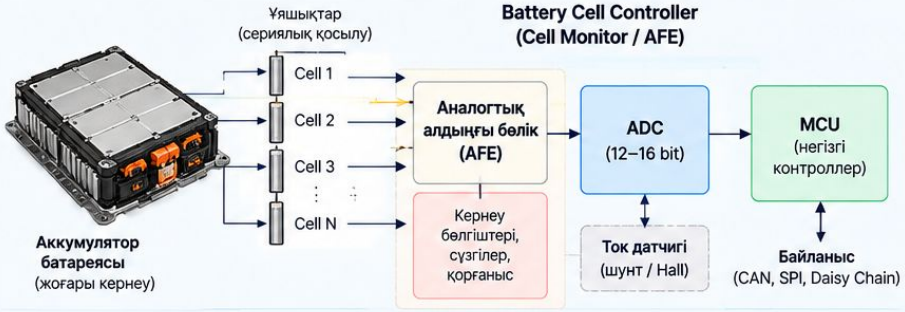
9 Нақты электромобильде маңызы

- Датчиктің дұрыс жұмысын тексеру
- Қауіпсіздік талаптарын орындау (ISO 26262)
- Қате сигналдардан қорғау
- Сенімді тарту күшін басқару
- Жүргізушінің қауіпсіздігін қамтамасыз ету

8. Электромобильдің BMS жүйесіндегі ADC

ADC технологиясы аккумуляторлық батареяның параметрлерін дәл өлшеуге мүмкіндік береді. BMS бұл мәліметтерді қауіпсіз басқару үшін қолданады.

1 BMS жүйесінің жалпы құрылымы



2 BMS жүйесінің негізгі міндеттері

- **Ұяшықтардың кернеуін өлшеу** — Әр ұяшықтың кернеуін дәл бақылау
- **Аккумулятор тогын өлшеу** — Заряд/разряд тогын анықтау
- **Температураны бақылау** — Ұяшықтар мен жүйенің температурасын өлшеу
- **Заряд деңгейін бағалау (SoC)** — Қалған энергия мөлшерін анықтау
- **Техникалық күйін бағалау (SoH)** — Батареяның қартаю деңгейін анықтау
- **Ұяшықтарды теңгеру (Cell Balancing)** — Ұяшықтардың заряд деңгейін теңестіру

3 ADC арқылы өлшенетін параметрлер

Ұяшық кернеуі



3.0 – 4.5 В
(әр ұяшық)

- Жоғары дәлдік (±1–5 мВ)
- Артық заряд/разряд қорғанысы

Ток өлшеу



Шунт / Hall
(±500 А ...)

- Заряд/разряд тогы
- Қуат есебі
- SoC есептеу үшін

Температура



NTC термистор
(–40 ... 125 °C)

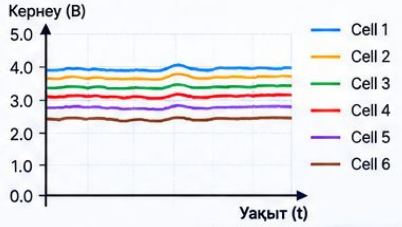
- Ұяшық температурасы
- Модуль температурасы
- Қорғаныс шектерін бақылау

Қосымша сигналдар

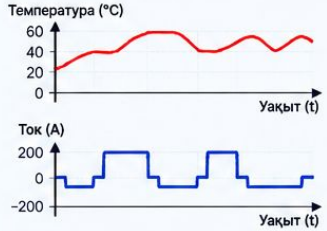


- Қорғаныс сигналдары
- Контактор күйі
- Изоляция бақылауы
- Сенсорлардың диагностикасы

4 Ұяшық кернеуін өлшеу мысалы



5 Температура мен токтың өлшенуі



6 ADC-дің маңызды рөлі

- Батарея параметрлерін жоғары дәлдікпен өлшейді.
- Қауіпсіздік шектерін бақылауға көмектеседі.
- Ұяшықтардың тепе-теңдігін қамтамасыз етеді.
- SoC және SoH есептеулерінің негізін қалайды.
- Ақаулықтарды анықтау және қорғаныс функцияларын іске қосады.

7 BMS микросхемаларының ерекшелігі



- Көбіне арнайы Battery Cell Controller немесе AFE микросхемалары қолданылады.
- Бірнеше ұяшықтың кернеуін бір мезгілде өлшейді (мысалы, 6–18 ұяшық).
- Жоғары дәлдікті ADC (12–16 бит).
- Кіріс сүзгілері, қорғау және диагностика функциялары бар.
- CAN немесе SPI арқылы негізгі MCU-мен байланысады.

8 Нәтижелердің қолданылуы

ADC өлшеу нәтижелері
(кернеу, ток, температура, т.б.)

↓

BMS алгоритмдері
(SoC, SoH, қорғаныс, теңгеру)

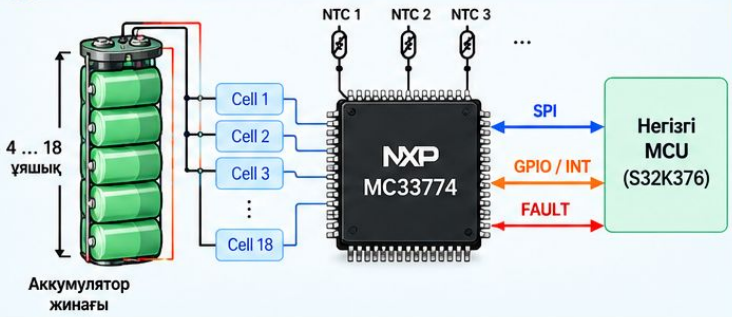
↓

Басқару командалары
(контакторлар, салқындату, зарядтау режимі, ескертулер)

9. NXP MC33774 – аккумулятор ұяшықтарының параметрлерін бақылау микросхемасы

NXP MC33774 — 4-тен 18-ге дейінгі аккумулятор ұяшықтарының кернеуін, температурасын және басқа параметрлерін өлшейтін, BMS жүйесіне арналған жоғары дәлдікті аналогтық-цифрлық өлшеу микросхемасы.

1 Жүйенің жалпы құрылымы



2 Негізгі мүмкіндіктері

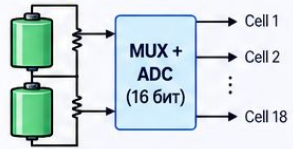
- 4 пен 18 ұяшыққа дейін (кернеу өлшеу)
- Сыртқы температура датчиктерін өлшеу (NTC)
- Өлшеу нәтижелерін цифрлық өңдеу (ADC)
- Аккумулятор ұяшықтарын теңгеру (Cell Balancing)
- Диагностикалық функциялар және ақаулықтарды анықтау
- SPI арқылы басқару және сандық байланыс

3 Негізгі техникалық сипаттамалары

Ұяшықтар саны	4 – 18
Ұяшық кернеуін өлшеу диапазоны	0 … 5.0 В
Өлшеу дәлдігі (ұяшық кернеуі)	±1–2 мВ (typ.)
Температура кірістері	3 NTC
ADC рұқсаты	16 бит
Байланыс интерфейсі	SPI, GPIO, INT, FAULT
Ұяшықтарды теңгеру	Пассивті теңгеру (сыртқы резисторлар)
Диагностикалық функциялар	Ашық / Қысқа тұйықталу, артық/кем кернеу, жоғары температура, және т.б.
Қауіпсіздік стандарты	ISO 26262 (міндеттерге сай)

4 Өлшеу арналары мен жұмыс принципі

Ұяшық кернеуін өлшеу



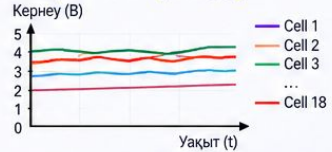
- Әр ұяшықтың кернеуі жеке өлшенеді
- Жоғары дәлдік: ±1–2 мВ

Температураны өлшеу



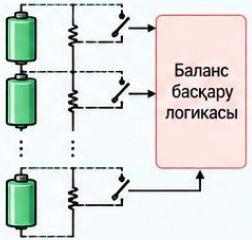
- 3 сыртқы температура датчигіне дейін
- Ұяшықтар мен модуль температурасын бақылау

Өлшеу нәтижелерін өңдеу



- ADC нәтижелерін цифрлық сүзгілеу
- Орташа мәнді есептеу
- VCU/MCU-ға жіберу

5 Ұяшықтарды теңгеру (Cell Balancing)



- Әр ұяшыққа бөлек теңгеру арнасы
- Пассивті теңгеру (сыртқы резисторлар арқылы)
- Ұяшықтар арасындағы кернеу айырмасын азайту
- Батареяның сыйымдылығының ұзақ мерзімді тұрақтылығын қамтамасыз ету

6 Диагностикалық функциялар

- Артық кернеу (Overvoltage)
- Кем кернеу (Undervoltage)
- Ашық тізбек (Open Circuit)
- Қысқа тұйықталу (Short Circuit)
- Жоғары температура (Overtemperature)
- Теңгеру ақаулары
- Өлшеу қателіктері
- Жалпы жүйелік ақаулықтар

7 Байланыс және интеграция



- Негізгі MCU (мысалы, NXP S32K376) арқылы SPI интерфейсімен мәліметтер алмасу
- Ақаулық сигналдарын INT/FAULT арқылы беру
- BMS/VCU бағдарламасында үздіксіз мониторинг және өңдеу

8 Қолдану салалары және стандарттар



ISO 26262

- Электромобильдердің аккумуляторлық батарея жүйелерінде
- Автомобильдік қауіпсіздік талаптарына сәйкес (ISO 26262)
- NXP S32K376 BMS/VCU платформасында MC33774 және MC33665 микросхемалары бірге қолданылады

Аккумулятордың температурасы — BMS бақылайтын негізгі параметрлердің бірі.

Температураны өлшеу үшін NTC терморезисторлары қолданылуы мүмкін.

NTC (Negative Temperature Coefficient) — температура жоғарылаған сайын электрлік кедергісі азаятын терморезистор.

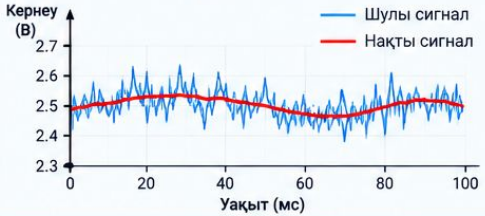
Аккумулятор тогын өлшеу – BMS аккумулятордан өтетін электр тогын да бақылайды.

13. Filtering – өлшеу нәтижелерін сүзгілеу

Нақты аналогтық сигналдарда электрлік шу мен кедергілер болуы мүмкін. Орташа мәнді есептеу (moving average) көмегімен ADC нәтижелерін сүзгілеуге болады.

1 Нақты сигналдардағы шу (кедергілер)

Датчиктің аналогтық сигналына электрлік шу, ЭМИ және басқа кедергілер әсер етуі мүмкін.



2 ADC нәтижелері (шулы деректер)

Физикалық шама өзгермеген жағдайда да ADC әртүрлі мәндер беруі мүмкін.

Өлшеу реті	1	2	3	4	5	6
ADC мәні	510	514	511	507	513	509

Арифметикалық орташа мән формуласы:

$$ADC_{avg} = \frac{ADC_1 + ADC_2 + \ldots + ADC_n}{n}$$

3 Мысал: орташа мәнді есептеу

Берілген мәндер үшін:

`510, 514, 511, 507, 513, 509`

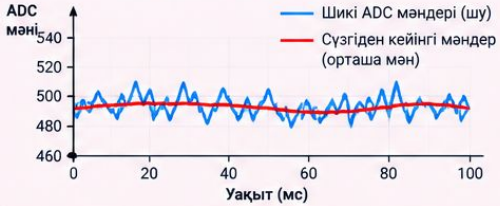
Орташа мән:

$$ADC_{avg} = \frac{510 + 514 + 511 + 507 + 513 + 509}{6}$$

$$ADC_{avg} = \frac{3064}{6} = 510.67 \approx \mathbf{511}$$

4 Сүзгілеудің әсері

Орташа мәнді есептеу шуларды азайтады және өлшеу тұрақтылығын арттырады.



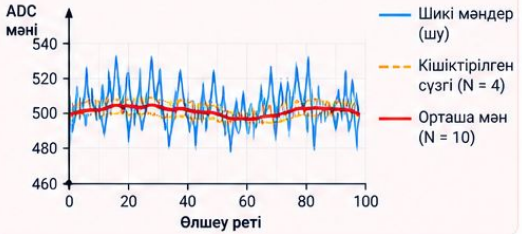
5 Бағдарламалық мысал (C++)

```cpp
float readAverage(int pin) {
  const int N = 10;        // өлшеу саны
  long sum = 0;

  for (int i = 0; i < N; i++) {
    sum += analogRead(pin);  // ADC мәнін оқу
    delay(1);                // тұрақтану уақыты (қажет болса)
  }

  return sum / (float)N;     // арифметикалық орташа мән
}
```

6 Сүзгілеу әдістерінің салыстырмасы



7 Қолданылатын сүзгілеу әдістері

Арифметикалық орташа мән (Moving Average)

$$y = \frac{1}{n}\sum_{i=1}^{n} x_i$$

- Жеңіл іске асыру
- Шуларды тиімді азайту
- Ең кең қолданылатын

Экспоненциалды сүзгі (EMA)

$$y_k = \alpha x_k + (1-\alpha) y_{k-1}$$

- Жылдам реакция
- Жадта бір ғана мән сақталады
- Көп жағдайда қолданылады

Медианалық сүзгі

$$y = \text{median}(x_1 \ldots x_n)$$

- Шұғыл ауытқуларды жою (outliers)
- Тұрақты нәтижелер
- Кедергілерге қарсы тұрақтылық

Сандық төменгі жиілікті сүзгі (LPF)

$$H(s) = \frac{1}{1 + sRC}$$

- Жоғары жиілікті шуды басады
- Аналогтық немесе сандық түрде іске асырылады

8 Практикалық ескертпелер

- Өлшеу санын (N) жүйенің жылдамдығына қарай таңдаңыз.
- Сүзгілеу ADC нәтижелерінің тұрақтылығын арттырады.
- Автомобильдік ортада ЭМИ және кедергілер жоғары, сондықтан сүзгілеу міндетті түрде қажет.
- Қажет болса, орташа мәнмен қатар медианалық немесе экспоненциалды сүзгілерді қолданыңыз.
- Сүзгілеу реакция уақытын ұлғайтуы мүмкін, бұл қауіпсіздік жүйелерінде ескерілуі тиіс.

```ino
const int pedal1Pin = PA0;
const int pedal2Pin = PA1;
const int faultLED = PB8;

// Оқу моделінің шартты параметрлері
const float V_MIN = 0.5;
const float V_MAX = 2.8;
const float MAX_DIFF = 8.0;

// ADC мәнін кернеуге түрлендіру
float readVoltage(int pin) {
  int adc = analogRead(pin);
  return adc * 3.3 / 4095.0;
}

// Кернеуді педаль пайызына түрлендіру
float pedalPercent(float voltage) {
  float percent =
      (voltage - V_MIN) /
      (V_MAX - V_MIN) * 100.0;
  return constrain(percent, 0.0f, 100.0f);
}

void setup() {
  Serial.begin(115200);

  analogReadResolution(12);

  pinMode(faultLED, OUTPUT);
}

void loop() {
  float voltage1 = readVoltage(pedal1Pin);
  float voltage2 = readVoltage(pedal2Pin);

  float pedal1 = pedalPercent(voltage1);
  float pedal2 = pedalPercent(voltage2);

  // Сигнал диапазондарын тексеру
  bool signalFault =
      voltage1 < V_MIN || voltage1 > V_MAX ||
      voltage2 < V_MIN || voltage2 > V_MAX;

  // Екі арнаның сәйкестігін тексеру
  bool mismatch =
      fabsf(pedal1 - pedal2) > MAX_DIFF;

  bool fault = signalFault || mismatch;

  digitalWrite(faultLED,
               fault ? HIGH : LOW);

  Serial.print("Pedal 1: ");
  Serial.print(pedal1);

  Serial.print("% | Pedal 2: ");
  Serial.print(pedal2);

  Serial.print("% | Status: ");
  Serial.println(fault ? "FAULT" : "NORMAL");

  delay(200);
}
```

Бақылау сұрақтары

1. ADC дегеніміз не?
2. Электромобильде ADC қандай мақсатта қолданылады?
3. 12 биттік ADC қанша цифрлық деңгейден тұрады?
4. ADC разрядтылығы мен дәлдігінің айырмашылығы қандай?
5. ADC кодын кернеуге қалай түрлендіруге болады?
6. APPS датчигі қандай қызмет атқарады?
7. Екі арналы APPS сигналдары не үшін салыстырылады?
8. NXP MC33774 қандай құрылғы?
9. NTC терморезисторының кедергісі температураға қалай тәуелді?
10. Filtering және Calibration процестерінің айырмашылығы қандай?
11. Plausibility Checking не үшін қажет?