10-ДӘРІС

Ендірілген жүйені жобалау және интеграциялау

Талаптар, модульдік архитектура, тестілеу және прототиптеу

әл-Фараби атындағы Қазақ ұлттық университеті

Физика-техникалық факультет

Дәрістің мақсаты және күтілетін нәтижелер

- Негізгі ұғымдарды түсіндіру: Ендірілген жүйе, Модульдік дизайн, Валидация.
- Raspberry Pi Pico құрылғысында тиісті MicroPython API құралдарын қолдану.
- Аппараттық қосылымның қауіпсіздігін және бағдарламаның дұрыстығын бағалау.
- Өлшеу немесе басқару алгоритмін тәжірибелік міндетке бейімдеу.

ТЕОРИЯ

Негізгі ұғымдар және анықтамалар

Ендірілген жүйе

Нақты функцияны орындайтын аппараттық-бағдарламалық кешен.

Модульдік дизайн

Жүйені тәуелсіз компоненттерге бөліп ұйымдастыру.

Валидация

Жүйенің пайдаланушы талабын қанағаттандыруын тексеру.

Теориялық негіздер: басты қағидалар

- Жобалау функционалдық талаптан басталады: өлшенетін шама, дәлдік, әрекет ету уақыты және қоректендіру.
- Датчик, логика, байланыс және атқарушы құрылғыны бөлек модульдерге ажырату тестілеуді жеңілдетеді.
- Fail-safe күйі, watchdog, қателерді тіркеу және аппараттық шектеулер жүйе сенімділігін арттырады.

АРХИТЕКТУРА

Жүйенің функционалдық құрылымы



Блок-схема сигналдың немесе деректердің негізгі қозғалыс бағытын көрсетеді. Нақты қосылым пиндері және қоректендіру талаптары қолданылатын модульдің техникалық құжаттамасымен салыстырылады.

ТАЛДАУ

Аппараттық және бағдарламалық байланыс

Аппараттық қабат

GPIO, қоректендіру, датчик, шина және физикалық сигнал деңгейлері.

Бағдарламалық қабат

MicroPython кітапханалары, алгоритм, деректерді өңдеу және қате жағдайлары.

Инженерлік қағида: бағдарламаның дұрыс орындалуы электрлік қосылымның қауіпсіздігін автоматты түрде қамтамасыз етпейді.

Датчикті оқу функциясы

```
from machine import ADC, Pin
sensor = ADC(Pin(26))
def read_voltage():
    return sensor.read_u16() * 3.3 / 65535
print(read_voltage())
```

Ескерту: кодты іске қоспас бұрын пиндер мен модуль түрін тексеріңіз.

Басқару логикасын бөлу

```
from machine import Pin
relay = Pin(15, Pin.OUT)
THRESHOLD = 2.0
def control(voltage):
    relay.value(1 if voltage > THRESHOLD else 0)
# Реле модулінің белсенді деңгейін тексеріңіз
```

Ескерту: кодты іске қоспас бұрын пиндер мен модуль түрін тексеріңіз.

Қателерді өңдеу

```
from time import sleep
while True:
    try:
        voltage = read_voltage()
        control(voltage)
    except Exception as err:
        relay.value(0)  # Қауіпсіз күй
        print("Қате:", err)
    sleep(1)
```

Ескерту: кодты іске қоспас бұрын пиндер мен модуль түрін тексеріңіз.

Практикалық қолдану сценарийлері

- Автоматты суару немесе температура мониторингі жобасының блок-схемасын әзірлеу.
- Қалыпты, шекті және ақаулы жағдайларға арналған тест-кейстер құру.
- Тәжірибе барысында алынған нәтижені кесте немесе қысқа техникалық есеп түрінде рәсімдеу.

Жиі кездесетін қателер және қауіпсіздік

- Аппараттық қорғаныссыз қуатты жүктемені басқару.
- Қателерді өңдемеу және талаптарды өлшенетін көрсеткіштермен жазбау.
- Құрылғыны жалғар алдында қорек көзін ажыратыңыз; 3,3 В логикалық деңгей шегін сақтаңыз.

БЕКІТУ

Бақылау сұрақтары және талқылау

- Верификация мен валидация айырмашылығы қандай?
- Fail-safe күйі қалай таңдалады?
- Модульдік құрылым тестілеуді қалай жеңілдетеді?

Қорытынды және ұсынылатын әдебиеттер

Дәріс бойынша қорытынды: теориялық ұғымдар MicroPython коды және аппараттық жүйе жұмысы арқылы бекітіледі.

- Raspberry Pi Ltd. Raspberry Pi Pico-series documentation: raspberrypi.com/documentation/microcontrollers/
- Raspberry Pi Ltd. RP2040 Datasheet: datasheets.raspberrypi.com/rp2040/rp2040-datasheet.pdf
- MicroPython documentation: docs.micropython.org/en/latest/rp2/quickref.html

- Қосымша: пайдаланылатын датчиктер мен модульдердің өндіруші дерекпарақтары.