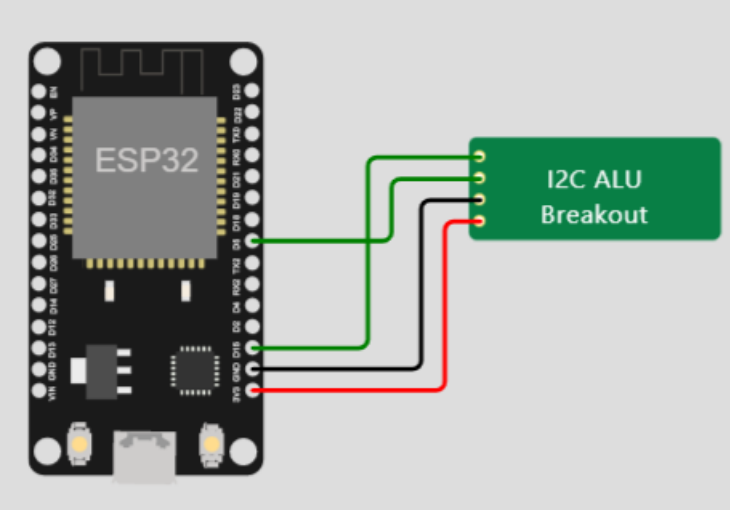
Programski zadatak 2 (10 bodova).

Na računalni sustav zasnovan sa ESP32 dodana je I2C Aritmetičko logička jedinica kako je prikazano na slici:



Potrebno je napisati program koji će koristiti I2C aritmetičko logičke jedinice za računanje hipotenuze pravokutnog trokuta:

$$c = \sqrt{a^2 - b^2}$$

Za računanje izraza nije dozvoljeno korištenje floar matematičkih operacija unutar c prevodioca već je potrebno koristiti I2C ALU sklop.

I2C ALU Sklop spojen je na sljedeći način:

Naziv signala	GPIO na ESP32
SCL - CLK	15
SDA	5

Ne predajete cijeli projekt nego predajete zip arhivu koja sadrži sve c/cpp i h datoteke koje ste kreirali

Za dobivanje prikazane sheme potrebno je unutar Wokwi projekta dodati dvije datoteke:

- i2c-alu.chip.c
- i2c-alu.chip.json

Zatim je potrebno modificirati *diagram.json* datoteku na sljedeći način:

```
{
  "version": 1,
  "author": "Sta god",
  "editor": "wokwi",
  "parts": [
    { "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": 0, "left": 0, "attrs": {} },
    { "type": "chip-i2c-alu", "id": "chip1", "top": 45.17, "left": 198.53, "attrs": {} }
  ],
  "connections": [
    [ "esp:TX0", "$serialMonitor:RX", "", [] ],
    [ "esp:RX0", "$serialMonitor:TX", "", [] ],
    [ "chip1:GND", "esp:GND.1", "black", [ "h-26.5", "v79.22" ] ],
    [ "chip1:VCC", "esp:3V3", "red", [ "h-15.83", "v-1.05" ] ],
    [ "chip1:SDA", "esp:D5", "green", [ "h-39.83", "v32.82" ] ],
    [ "esp:D15", "chip1:SCL", "green", [ "h51.53", "v-83.33" ] ]
  ]
}
```

Sve tri datoteke nalaze se unutar zip arhive uz tekst zadatka.

I2C ALU sklop:

I2C ALU sklop je „slave“ I2C uređaj koji se nalazi na **adresi 0x22** i obavlja operaciju sa brojevima u float formatu.

Interno sklop ima tri registra u koje je moguće pisati, čitanje iz registara nje moguće. Čitanjem sa ALU sklopa dobiva se rezultat operacije. Pogledati (**ČITANJE SA ALU I2C SKLOPA).**

Upis u registre se mora obavljati jedan po jedan. Nije moguće odjednom upisati podatke u više registara.

REGISTAR OPERACIJE (0x00)

Registar operacija je veličine jednog bajta koji određuje koja operacija će se raditi. Popis operacija je sljedeći:

Operacija	Vrijednost	Opis operacije
OPERATION_NOP	0x00	ne radi ništa ali kod čitanja vraća -0.101010
OPERATION_MUL	0x01	množi podatke spremljene u registre PAR1 i PAR2
OPERATION_DIV	0x02	dijeli registrima PAR1 sa registrom PAR2 (PAR1/PAR2), ako je PAR2 nula vraća nulu (0x00).
OPERATION_POWER_A	0x03	računa kvadrat od PAR1
OPERATION_POWER_B	0x04	računa kvadrat od PAR2
OPERATION_SQRT_A	0x05	računa drugi korijen od PAR1
OPERATION_SQRT_B	0x06	računa drugi korijen od PAR2
OPERATION_READ_A	0x64	Ne računa ništa, samo vraća PAR1
OPERATION_READ_B	0x65	Ne računa ništa, samo vraća PAR2

REGISTAR PAR1 (0x01)
Pisanjem u ovaj registar postavlja se vrijednost prvog parametra (PAR1). Registar je tipa float i za upisivanje je potrebno poslati 4 bajta koja predstavljaju float broj. Prvo se šalje niži bajt i onda prema višim bajtovima.

REGISTAR PAR2 (0x01)
Pisanjem u ovaj registar postavlja se vrijednost drugog parametra (PAR2). Registar je tipa float i za upisivanje je potrebno poslati 4 bajta koja predstavljaju float broj. Prvo se šalje niži bajt i onda prema višim bajtovima.

Za pretvorbu float broja u bajt polje najjednostavnije je koristiti union strukturu:

```
union floatunion_t {
   float f;
   unsigned char a[sizeof (float) ];
};
```

Iz ove union strukture jednostavno možete čitati i pisati float broj i bajt polje.

ČITANJE SA ALU I2C SKLOPA
Prilikom čitanja obavlja se operacija koja je zadana u internom registru operacija i potrebno je pročitati 4 bajta. Pročitana 4 bajta predstavljaju rezultat u float formatu. Prvo se čita niži bajt i tako redom do najvišeg.

Primjer pseudo koda slanja parametara za računanja operacije zbrajanja:

INIT_I2C
SEND_I2C (0x00, 0x01) šalji operaciju 1,
SEND_I2C (0x01, 0xRR, 0xTT, 0xUU, 0xHH) pošalji broj 0xHHUUTTRR koji predstavlja neki float broj PAR1
SEND_I2C (0x02, 0xRR, 0xTT, 0xUU, 0xHH) pošalji broj 0xHHUUTTRR koji predstavlja neki float broj PAR2
READ_I2C(4 bajta) -> pročitana četiri bajta su rezultat operacije

Predefinirane vrijednosti:

```
#define CONFIG_SCL_GPIO       15
#define CONFIG_SDA_GPIO       5
#define ALU_ADDR 0x22
#define ALU_OPER      0
#define ALU_PAR1      1
#define ALU_PAR2      2
#define OPERATION_NOP  0
#define OPERATION_ADD  1
#define OPERATION_SUB  2
#define OPERATION_MUL  3
#define OPERATION_DIV  4
#define OPERATION_POWER_A 5
#define OPERATION_POWER_B 6
#define OPERATION_SQRT_A 7
#define OPERATION_SQRT_B 8
#define OPERATION_READ_A 100
#define OPERATION_READ_B 101
```