Vaja 13
RTOS-2

1. **S pomočjo izpopolnjenega operacijskega sistema v realnem času (RTOS2) ponovite vajo 12**
2. **Periodične naloge (tipke, lcd, uart) naj se izvajajo kot opravila, glavni program (funkcija main()) naj skrbi za komunikacijo med različnimi enotami.**

RTOS2 (rtos2.h, rtos2.c, rtos2mem.c, rtos2pipe.c, rtos2sched.c, rtos2task.c) :

1. **opravila se razvrščajo na podlagi prioritet**
2. **časovni interval, v katerem je opravilo enkrat na vrsti, se določa za vsako opravilo posebej**
3. **izvajanje opravila ni časovno omejeno, čeprav morajo biti opravila končno dolga**
4. **prekinitve delujejo neovirano, razen prekinitve, ki jo uporablja operacijski sistem (timer0 je prekinitev z najvišjo prioriteto)**
5. **dinamično dodajanje in odstranjevanje opravil, podatkovnih cevi**
6. **dinamično dodeljevanje pomnilnika**

Q: Kakšne naj bodo prioritet opravil
Q: Kakšne naj bodo frekvence izvajanja opravil
Q: Ali je sistem razvrstljiv

Inicializacija (ne zagon):

*void rtos2init(char *memory, unsigned int size, unsigned int slice);*

memory ... naslov začetka sistemskega dinamičnega pomnilnika
(globalno polje določene velikosti)
size ... velikost sistemskega dinamičnega pomnilnika v bajtih
(odvisna od št. opravil in št. in velikosti podatkovnih cevi)
slice ... dolžina časovne rezine v mikrosekundah

Nastavi timer0 vektorsko prekinitev z najvišjo prioriteto =>

- **ostale prekinitve je potrebno nastaviti prej**
- *vic_init(…)* **kličemo samo pred klicem** *rtos2init(…)*
- **vektorske prekinitve 0 ne moremo uporabiti**

Zagon in ustavitev:

void enable_os();
void disable_os();

Ko je rtos2 ustavljen ga lahko ponovno inicializiramo (zahteva ponovni inicializacijo opravil)

Primer zagona rtos2:

```
// Globalno polje s 500 bajti sistemskega pomnilnika.
char system_memory[500];
...

// Nastavitev prekinitev. Razen vektorske IRQ z najvišjo prioriteto in prekinitve timer0.
vic_init(...);

// Inicializacija RTOS2 s časovno rezino 10ms in 500 Byte-i sistemskega pomnilnika
rtos2init(system_memory, 500, 10000);

// Dvig operacijskega sistema.
enable_os();
...

// Ustavitev operacijskega sistema.
disable_os();
```

Podatkovne cevi (medpomnilniki):

- uporabljamo v glavnem programu in opravilih

- za pisanje / branje v drugih prekinitvah je potrebno začasno onemogočiti prekinitve (disable_intr(…), enable_intr(…))

Primer uporabe:

```
// prostor za podatke
char buf[10];
unsigned int n;

// kazalec na podatkovno cev
struct rtos2pipe *transfer;
  ...
// Ustvarimo cev z medpomnilnikom velikosti 100 bajtov.
transfer = rtos2pipe_create(100);
  ...
// Poskus zapisa 10-ih bajtov iz buf. n=št. uspešno zapisanih bajtov
n = rtos2pipe_write(transfer, buf, 10);
  ...
// Poskus branja 10-ih bajtov v buf. n=št. uspešno prebranih bajtov
n = rtos2pipe_read(transfer, buf, 10);
  ...
// Uničimo cev.
rtos2pipe_delete(transfer);
```

Opravila:

```
typedef void (* rtos2taskptr)(void *);     // kazalec na funkcijo opravila
void rtos2task_create(rtos2taskptr function, void *arg, unsigned int priority,
                      unsigned int interval);
```

function	...	*funkcija opravila,*
arg	...	*argument s katerim je opravilo klicano (0, če ga ne uporablja)*
priority	...	*prioriteta opravila (nižja vrednost pomeni višjo prioriteto)*
interval	...	*število časovnih rezin, ki sestavljajo interval v katerem je opravilo enkrat na vrsti.*

Primer uporabe:

```
// prostor za parametre, ki jih želimo podati ob klicu opravila
char arg[10];
// funkcija opravila
void task(void *data);
  ...
// Opravilo task(arg) bo na vrsti v vsaki 100-ti rezini.
rtos2task_create(task, arg, 5, 100);
  ...
// Uničimo opravilo.
rtos2task_delete(task);
```

Vaja 13:

<u>Opravila:</u>

1. *void keys_driver(void *arg);*

2. *void lcd_driver(void *arg);*

3. *void uart_driver(void *arg);*

<u>Inicializacijske funkcije za opravila RTOS2:</u>

inicializirajo pine, registre => klic xxx_init() za xxx periferno enoto
ustvarijo medpomnilnike, če so potrebni => klic rtos2pipe_create()
ustavrijo opravilo => klic rtos2task_create()

1. *void rtos2_keysInit(); // ustvari medpomnilnik, nastavi opravilo*

2. *void rtos2_lcdInit(); // nastavi opravilo*

3. *void rtos2_uartInit(); // ustvari RX in TX medpomnilnik, nastavi opravilo*