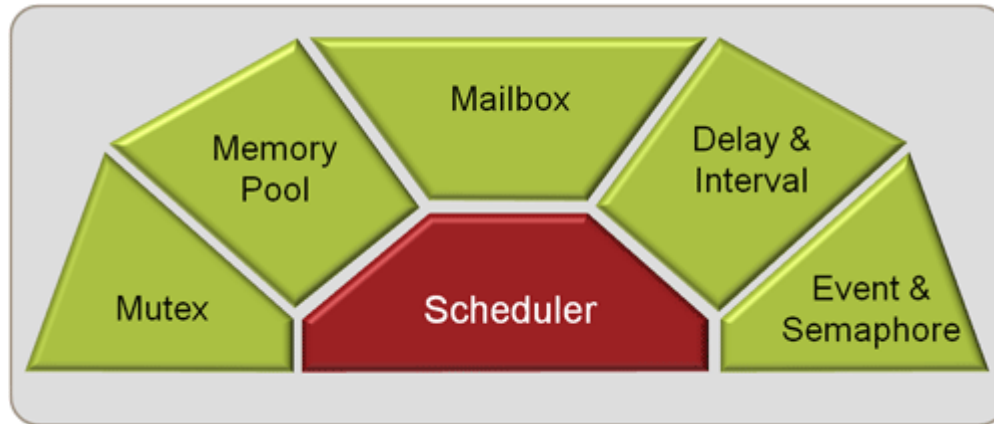


RTOS koncept

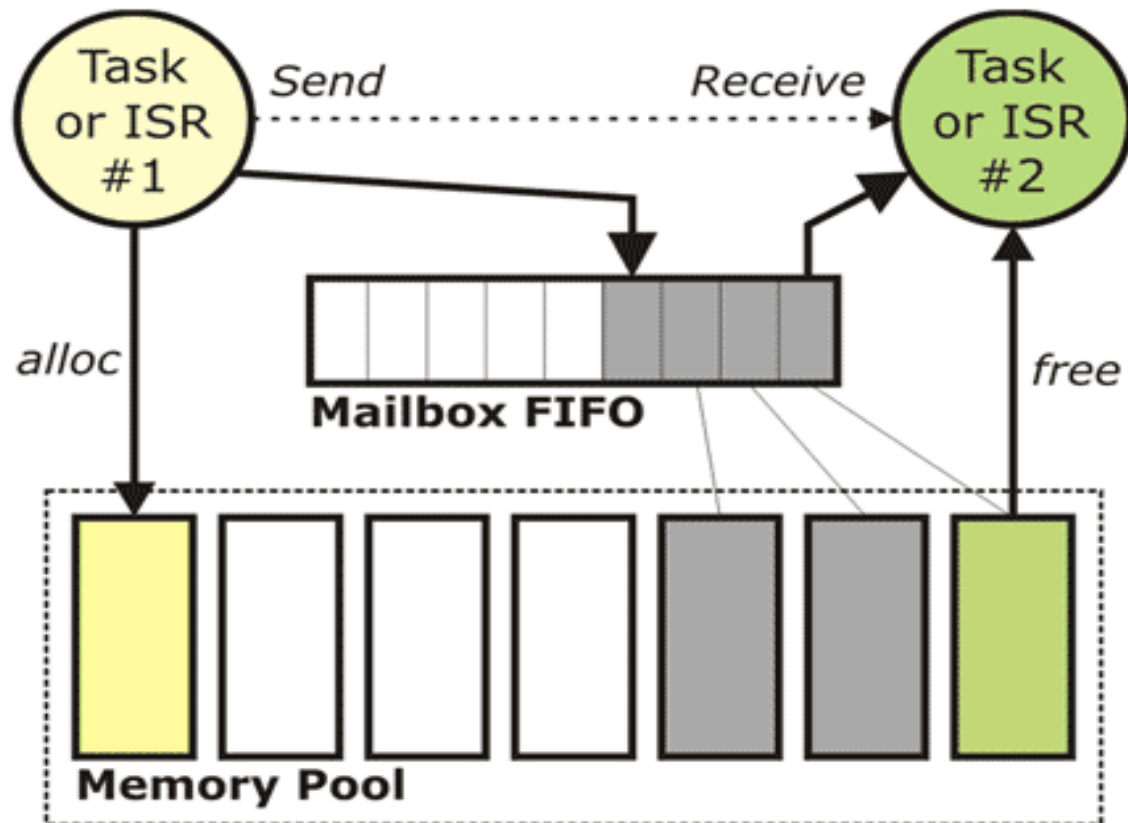


- Jednoduché vestavěné systémy jsou ovládány pomocí **super smyčky**.
- Složité systémy používají Real-Time Operating System (RTOS).
- Jednotlivé programové funkce tvoří samostatné úlohy (ang. **tasks**)
- Provádění úloh řídí krátkodobý plánovač (ang. **scheduler**)

Výhody RTOS, jak např. **Keil RTX**

- **Task scheduling** - tasks are called when needed ensuring better program flow and event response
- **Multitasking** - task scheduling gives the illusion of executing a number of tasks simultaneously
- **Deterministic behaviour** - events and interrupts are handled within a defined time
- **Shorter ISRs** - enables more deterministic interrupt behaviour
- **Inter-task communication** - manages the sharing of data, memory, and hardware resources among multiple tasks
- **Defined stack usage** - each task is allocated a defined stack space, enabling predictable memory usage
- **System management** - allows you to focus on application development rather than resource management (housekeeping)

Úlohy 1 & 2 jsou funkce v aplikaci. RTX řídí přidělování paměti i mailboxu tak jak probíhá komunikace mezi tasky č.1 a č.2.




RTX používá jednoduchou syntax pro přímý přístup ke všem zdrojům RTOS

```
os_mbx_declare (mailbox1, 20);

__task void task1 (void) {
    void *msg;

    os_mbx_init (mailbox1, sizeof(mailbox1));
    msg = alloc();
    // fill message content

    os_mbx_send (mailbox1, msg, 12);
}
```



```
__task void task2 (void) {
    void *msg;

    ..
    os_mbx_wait (mailbox1, &msg, 100);
    // process message content here
    free (msg);
}
```

