RECAP SW 7
HOOKS, HEAP & SHELL

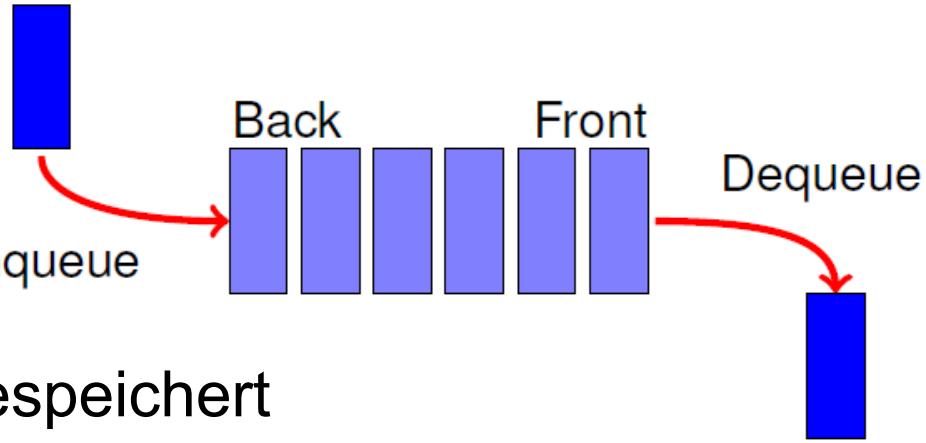
Robert Näger
Tobias Heer

10. November 2016

Ablauf

- Queues
- Hooks
- Shell

Prinzip von Queue



- First in / First out (FIFO)
- Statische Liste, in Heap gespeichert
- Operationen: Erstellen, Löschen, Elemente anfügen und auslesen
- Items by Value, not by reference

Hooks (Haken)

- Sind implementierte Callbacks, die vom RTOS bei Ereignisse aufgerufen werden
- 4 Hooks verfügbar: Idle Hook, Tick Hook, Malloc Failed Hook, Stack Overflow Hook
- Einschalten in FRTOSConfig.h im Ordner Generated_Code

Idle Hook

- Wird aufgerufen, wenn kein Task abgearbeitet wird.
- Ideal for low-power mode
- Darf keine API Funktionen aufrufen

Tick Hook

- Von Tick interrupt aufgerufen
- Kann als Alternative zum Timer verwendet werden
- Trotzdem kurz halt, da von Interrupts aufgerufen wird
- Nur von FromISR() RTOS API Funktionen können aufgerufen werden

Malloc Failed Hook

- Aufgerufen wenn memory allocation nicht erfolgreich war.
- Nicht genügend Speicher im Heap
- Evtl. Heap Size erhöhen

Stack overflow Failed Hook

- Stack hat nicht genügend Speicher
- Stack Pointer bei Taskwechsel überprüfen
- SP überprüfen und letzte 16 Bytes pattern überprüfen

Zusammenfassung

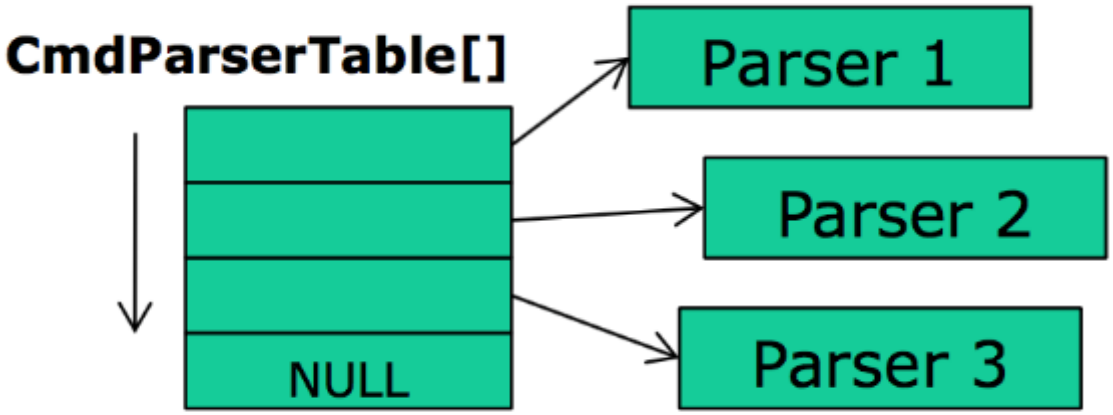
- Hooks werden als Events verwendet um Application zu informieren
- Optional
- 4 Hooks in FRTOS Komponente

Shell

```
static void ShellTask(void *pvParameters) {
  static uint8_t localConsole_buf[48];
  CLS1_ConstStdIOTypePtr ioLocal = CLS1_GetStdio();

  localConsole_buf[0] = '\0';
  (void)CLS1_ParseWithCommandTable(CLS1_CMD_HELP,
      ioLocal, CmdParserTable);
  for(;;) {
    (void)CLS1_ReadAndParseWithCommandTable(localConsole_buf,
        sizeof(localConsole_buf), ioLocal, CmdParserTable);
    FRTOS1_vTaskDelay(50/portTICK_RATE_MS);
  } /* for */
}
```

Parser Table




```
static const CLS1_ParseCommandCallback CmdParserTable[] =
{
  CLS1_ParseCommand, /* Processor Expert Shell component, is first in list */
  SHELL_ParseCommand, /* our own module parser */
#if FRTOS1_PARSE_COMMAND_ENABLED
  FRTOS1_ParseCommand, /* FreeRTOS shell parser */
#endif
```

Nützliche Funktionen

- UTIL1_strcmp() : vergleicht zwei Strings, Rückgabewert = 0 falls identisch
- UTIL1_xatoi() : ascii to int (binary, octal, decimal, hexadecimal)

Parser

```
static uint8_t SHELL_ParseCommand(const unsigned char *cmd, bool *handled, const CLS1_StdIOType *io) {
  uint32_t val;
  const unsigned char *p;

  if() {
  //............
  }
  else if (UTIL1_strncmp(cmd, "Shell val ", sizeof("Shell val ")-1)==0) {
    p = cmd+sizeof("Shell val ")-1;
    if (UTIL1_xatoi(&p, &val)==ERR_OK) {
      SHELL_val = val;
      *handled = TRUE;
    } else {
      return ERR_FAILED; /* wrong format of command? */
    }
  }
  return ERR_OK;
}
```

Quiz Queue

- Q: Wann muss die Grösse der Queue bekannt sein?

Quiz Queue

- Q: Wann muss die Grösse der Queue bekannt sein?

- A: Zur Kompilierzeit. Kann während Laufzeit nicht verändert werden.

Quiz Hook

- Q: Was sind Hooks? Und wieviele gibt es?

Quiz Hook

- Q: Was sind Hooks? Und wieviele gibt es?

- A: Callbacks in FRTOS, die bei Ereignisse (bsp. StackOF) aufgerufen werden. 4 Hooks.

Quiz Shell

- Q:

  Obwohl Hillary sonst nicht sehr IT-begabt ist, hat sie es geschafft ihren Shell-Parser so zu erweitern, dass die Eingabe von “result” zur Ausgabe des Strings “disappointing” führt. Nun möchte sie diese Ausgabe als automatische Willkommensnachricht bei jedem Systemstart in der Konsole zu sehen bekommen.

  Welche der beiden folgenden Funktionen benutzt sie dafür im ShellTask und warum?

  CLS1_ParseWithCommandTable()

  oder

  CLS1_ReadAndParseWithCommandTable()

Quiz Shell

- A:

  CLS1_ParseWithCommandTable()

  Kein Auslesen der Konsolen-Eingabe, kein Buffer notwendig, nicht im for-loop des ShellTasks.

Quiz Shell

- Q:

  Was ist der Unterschied zwischen den Funktionen UTIL1_strcmp() und UTIL1_strncmp() ?

Quiz Shell

- Q:

  Was ist der Unterschied zwischen den Funktionen UTIL1_strcmp() und UTIL1_strncmp() ?

- A:

  UTIL1_str**n**cmp() hat zusätzlich zu den zwei Strings ein size Parameter, der angibt bis zu welcher Position verglichen werden soll.