jocktos-docs

Release v0.1

Nick Schneider, Joshua Goard

Mar 07, 2024

CONTENTS:

A journey of learning how to create an RTOS kernel from the ground up, by Nick Schneider and Joshua Goard

CHAPTER

ONE

SOURCE CODE!

1.1 Embedded LaTeX Equations

Here's some LaTeX inside the reStructured-Text:

$$\frac{\sum_{t=0}^{N} f(t,k)}{N}$$

1.2 Data Structure Reference

Here's an auto-gen blurb on a coreRegisters data structure:

struct **coreRegistersDef**

STM32 Cortex-M4 Core Registers.

Public Members

uint32_t **registers**[13]

R0-R12 are 32-bit general-purpose registers for data operations.

uint32_t ***stack_pointer**

The Stack Pointer (SP) is register R13.

In Thread mode, bit[1] of the CONTROL register indicates the stack pointer to use:

- 0: Main Stack Pointer (MSP). This is the reset value.
- 1: Process Stack Pointer (PSP). On reset, the processor loads the MSP with the value from address 0x00000000.

uint32_t **link_register**

The Link Register (LR) is register R14.

It stores the return information for subroutines, function calls, and exceptions. On reset, the processor loads the LR value 0xFFFFFFFF.

uint32_t ***program_counter**

The Program Counter (PC) is register R15.

It contains the current program address. On reset, the processor loads the PC with the value of the reset vector, which is at address 0x00000004. Bit[0] of the value is loaded into the EPSR T-bit at reset and must be 1.

uint32_t **program_status_register**

The Program Status Register (PSR) combines:

- Application Program Status Register (APSR)
- Interrupt Program Status Register (IPSR)
- Execution Program Status Register (EPSR)

uint32_t **exception_markers**[3]

Exception Markers contain:

- Priority Fault Mask Register (PRIMASK)
- Fault Mask Register (FAULTMASK)
- Base Priority Mask Register (BASEPRI)

uint32_t **control_register**

The CONTROL register controls the stack used and the privilege level for software execution when the processor is in Thread mode and indicates whether the FPU state is active.

1.3 Enum Reference

Here's an auto-gen blurb on the task state enumeration:

enum **jocktos_TaskState**

Task State Machine Enumeration.

Attention

Wild inline LaTeX

$$\int_a^b f(x)dx = F(b) - F(a)$$

Warning: super janky image:

```
                  -------------
   /-->----->---->| SUSPENDED |<-----<-----<--\
  /               -------------                \
 /                     ^    v                   \
 /          suspend(); |   | resume();             |
|                      ^   v                       ^
^                    ---------   scheduler   -----------
| suspend();         | READY | >----->-----> | RUNNING |
^                    |       | <-----<-----< |         |
|                    ---------               -----------
 \                       ^                         v
  \                      | event                   |
   \                     ^                        /
    \               -----------                  /
     \<-----<-----<-| BLOCKED |<-----<-----<---/
                    -----------
```

Values:

enumerator **RUNNING**

Currently active task.

enumerator **READY**

In the queue and ready to run.

enumerator **BLOCKED**

Awaiting a resource.

enumerator **SUSPENDED**

Delayed or intentionally released.

This was all in the header file!

The figure and LaTeX can both be found in jocktos/main.h

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search

INDEX