Laser gaming

*Een oorlog zonder gevolgen*



|  |  |  |  |
| --- | --- | --- | --- |
| **Projectleden** |  | René de Kluis:  Brent Cranko:  Marten Butter:  Arco Gelderblom: | 1661627  1674186  1691576  1679531 |
| **Docent** |  | Marten Wensink  Jan Zuurbier |  |
| **Klas** |  | TI-ICT-V2B |  |
| **versie** |  | 1.0 |  |
| **Datum** |  | 25-10-2016 |  |

# Introduction

* Research question
* Available RTOS functions

*Hoodfvraag:*

‘Met behulp van welk open source realtime operating system kunnen tasks en de concurrency mechanismen, pool, channel, flag(group), clock timer en mutex, zoals aangeboden door het Arduino RTOS, met zo weinig mogelijk overhead worden gerealiseerd?’

*Deelvragen:*

1. Wat zijn de kenmerkende eigenschappen van tasks en de concurrency mechanismen van het Arduino RTOS?

Belangrijke kenmerken van de tasks van het Arduino RTOS zijn dat elke task een boolean state heeft. Dit betekend dat het bijvoorbeeld de statussen suspended/resumed of waiting/non-waiting kan zijn.

Ook is het bij tasks zo dat ze allemaal draaien in hetzelfde geheugenadres op de heap. Als dit namelijk niet zo zou zijn, bestaat er de kans dat bepaalde dingen van andere tasks veranderd worden. Als dit zou gebeuren zou de functionaliteit van die tasks niet meer betrouwbaar zijn.

Daarentegen heeft elke task wel zijn eigen stack, deze heeft een standard grote van 8Kb.

Daarnaast kan een task pas beginnen als de statussen op resumed en non-waiting staan. Als één van deze niet goed staat is het niet mogelijk voor de task om te beginnen.

Verder heeft elke task zijn eigen standard prioriteit, er zal dus ook geen andere task zijn met dezelfde prioriteit. Deze prioriteit is standard een unsigned waarde onder de 1000. Dit is “random” gekozen, maar eventueel kan deze ook ingesteld worden door de programmeur.

* + - Every task has 2 bi-value(boolean) states. (Suspended/resumed)(Waiting/non-waiting)
      * *Elke task heeft een boolean state. Bijvoorbeeld: suspended/resumed; waiting/non-waiting*
    - Each task is only runable if both flags are set to resumed and non-waiting.
      * *Elke task kan pas draaien als de beide flags van resumed en non-waiting gezet zijn.*
    - Each task runs in the same memory space.(wild pointers can ruin things in the other task)
      * *Elke task draait in dezelfde geheugenplaats. Als dit namelijk niet zo is kan dit dingen in andere tasks verknallen.*
    - each task has a fixed priority(unsigned value beneath 1000)
      * *Elke task heeft een standard prioriteit. Dit is unsigned value onder 1000.*
      * *Deze prioriteit kan worden aangepast*
    - priority's need to be singular. so no tasks with the same priority
      * *priotiteiten zijn uniek, dus er zijn geen tasks met dezelfde prioriteit.*
    - Each thread has it's own stack. task stack default size is 8Kb.
      * *Elke thread zit op zijn eigen stack. De standard grote voor deze stack is 8Kb.*

1. Welke open source RTOS-en zijn beschikbaar?
2. Welk van de beschikbare RTOS-en biedt de meeste van de concurrency mechanismen van het Arduino RTOS aan zonder enige modificatie en biedt dezelfde functionaliteit om taken te realiseren?
3. Welke mechanismen van het Arduino RTOS worden niet ondersteund door de beschikbare RTOS-en?
4. Hoe kunnen de mechanismen van het Arduino RTOS die niet direct worden ondersteund door de beschikbare RTOS-en worden gerealiseerd m.b.v. van deze RTOS-en?

# Inhoudsopgave

Inhoudsopgave

[Introduction 2](#_Toc465175408)

[Inhoudsopgave 7](#_Toc465175409)

[Research methods 8](#_Toc465175410)

[Results 9](#_Toc465175411)

[Conclusion 13](#_Toc465175412)

[Discussion 14](#_Toc465175413)

[References 15](#_Toc465175414)

# Research methods

* Criteria RTOS
  + Make a list based on the criteria given by our customer.
* Functions arduino RTOS
  + Literature study of the DoxyGen from the Arduino RTOS. Make a list of all available functionality.
* Alternatives for functionality arduino RTOS
  + Literature study of available functionalities and alternative implementations.

# Results

## List of RTOS-en

EcosPro

uKOS

BeRTOS

CoActionOS

cocoOS

Distortos

FreeRTOS

FunkOS

scmRTOS

Nut/OS

NuttX

[mbed OS](http://www.osrtos.com/rtos/mbed-os)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | License | Platforms | Version | Links | Last updated |
| [mbed OS](http://www.osrtos.com/rtos/mbed-os) | Apache License | ARM | 5.1 | [GitHub](https://github.com/ARMmbed/mbed-os) | 2016-Sep-17 |
| [StratifyOS](https://stratifylabs.co/) | GPL | ARM | v2.0.0-beta2 | [GitHub](https://github.com/StratifyLabs/StratifyOS) | 2016-Jul-21 |
| [distortos](http://distortos.org/) | Mozilla Public License Version 2.0 | ARM | v0.2.0 | [GitHub](https://github.com/DISTORTEC/distortos) | 2016-May-04 |
|  |  |  |  |  |  |

[StratifyOS](https://stratifylabs.co/)

## List of functionalities

* + Every task has 2 bi-value(boolean) states. (Suspended/resumed)(Waiting/non-waiting)
  + Each task is only runable if both flags are set to resumed and non-waiting.
  + each task runs in the same memory space.(wild pointers can ruin things in the other task)
  + each task has a fixed priority(unsigned value beneath 1000)
  + priority's need to be singular. so no tasks with the same priority
  + Each thread has it's own stack. task stack default size is 8Kb.
  + Has mutex
  + Has mailbox
  + Has timer
  + Has flags
  + Has pool
  + Has channel
  + Has event
  + Has waitables
  + All objects from RTOS can be printed to the ostream using << operator.
  + Printing the RTOS will print all RTOS objects.
  + “Trace <<” for debugging, vertraagd wel de tasks
  + The RTOS uses non-preemptive task switching.
    - This means that the CPU can be switched to another task only when the current executing task (directly or indirectly) calls an RTOS function.
* Can release task while running
* Tasks can make other tasks runnable
* Order of priorities have to be given, otherways the priority is “random”

## Matrix comparison

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **RTOS naam** | Ecos Pro | uKOS | Coco OS | Distort OS | FreeRT OS | Funk OS | scmRT OS | NuttX | Stratify OS | Mbed OS | School RTOS |
| **mutex** | V | V | V | V | V | V | V | V | V | V | V |
| **event flag** | V | V | V | V | V | V | V | V | V | V | V |
| **mailbox** | V | V | X | X | V | X | V | V | X | V | V |
| **channel** | X | V | V | X | X | X | V | V | X | X | V |
| **timer** | V | V | V | V | V | V | V | V | V | V | V |
| **waitables** | X | V | V | V | X | X | V | V | X | V | V |
| **pool** | X | X | V | X | X | X | V | V | X | X | V |
| **task** | X | X | V | X | V | V | V | V | X | V | V |
| **Message queue** | V | V | V | X | V | V | V | V | V | V | V |
| **semaphore** | V | V | V | V | V | V | V | V | V | V | V |
| **Threads** | V | V | X | V | X | V | V | V | V | X | V |
| **Spinlock** | V | X | X | X | X | X | X | V | X | X | X |
| **Round robin** | V | X | V | X | X | V | V | V | X | V | X |
| **Scheduler** | V | V | V | V | V | V | V | V | V | X | V |

## Missende functionaliteiten in sommige RTOS-en:

* Message queue 1
* Scheduler 2
* Waitables 3
* Threads 3
* Mailbox 4
* Tasks 4
* Channel 5
* Round robbin 6
* Pool 7
* Spinlock 9

## Oplossingen:

**Mailbox:**

* Zet een waarde in de queue
* Lock de queue door mutex
* Lees waarde uit de queue
* Unlock queue door mutex

**Channel:**

* Maak een message queue aan
* Zit in de 1e waarde de klasse naam
* De andere klasse zoekt naar de queue waarvan de 1e waarde de naam van de klasse is waarmee het communiceert

**Waitables:**

* Lock proces met mutex
* Zet timer op meegegeven waarde
* Unlock als timer algelopen is

**Pool:**

**Task:**

**Message queue:**

**Threads:**

**Spinlock:**

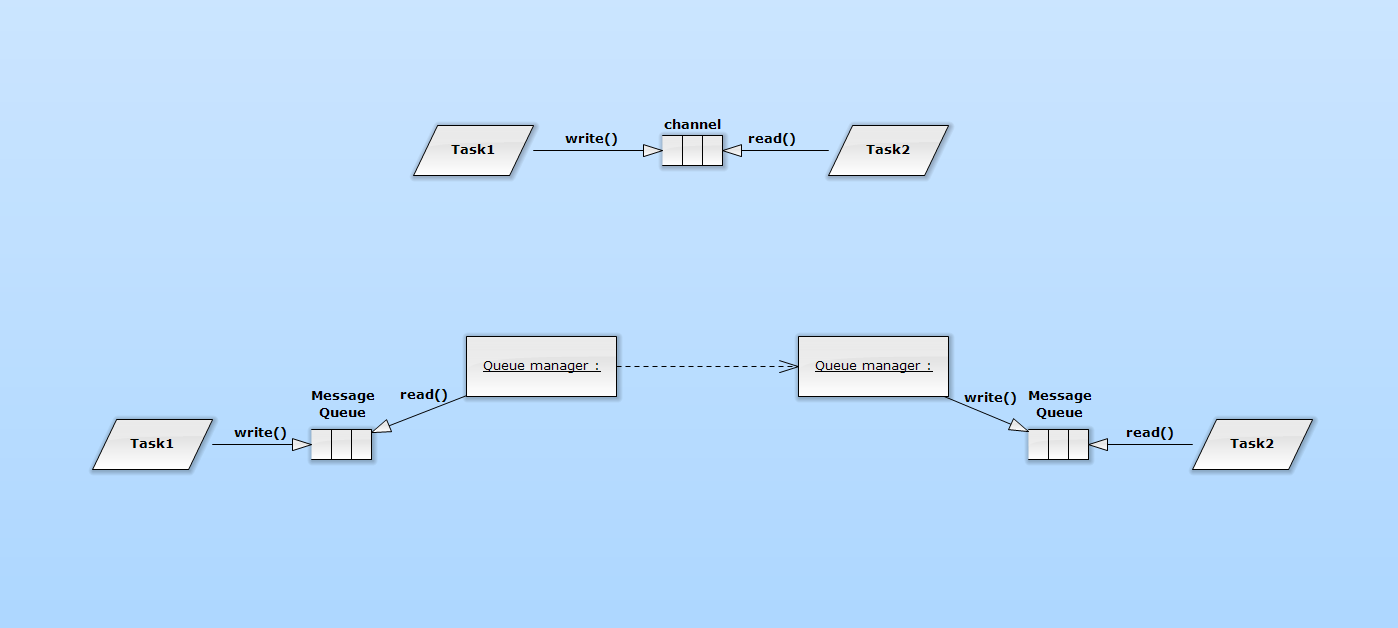
* Proces gaat in loop als er een event flag is ontvangen
* Proces blijft in loop totdat de event flag resume ontvangen word.

**Round robin:**

* Maak scheduler aan
  + Parameters:
    - queue met processen
    - interval
    - event flag suspended
    - event flag resume
* Timer in de scheduler afgelopen
  + Huidig proces word suspended door event flag
  + Volgend proces word gestart
* Start recursief opnieuw

**Scheduler:**

* Maak timer aan
  + Parameters:
    - Proces
    - Tijd
* Start timer
* Timer afgelopen
  + Communiceer met meegegeven proces
* Start recursief opnieuw



* Alternatives for functionality not available

Functionality 1

Functionality 3

Functionality 5

Etc

# Conclusion

* Best alternatives

# Discussion

* Other alternatives
* Faults in research

# References

**Open-source RTOS-en van:**

http://www.osrtos.com/

**stack vs heap:**

http://gribblelab.org/CBootcamp/7\_Memory\_Stack\_vs\_Heap.html