# Inleiding

Het ‘internet of things’ is een echte rage geworden. Tegenwoordig is alles verbonden met het internet, de koelkast, een wasmachine en zelfs verlichting. Omdat het internet altijd aan staat en je als apparaat eigenlijk altijd online moet zijn is een embedded systeem vaak een goede optie. Zo’n embedded systeem, voor een koelkast bijvoorbeeld, moet naast de connectie tot het internet ook nog andere dingen kunnen doen.

Voor onze opdracht moeten we een vergelijkbaar systeem maken maar dan voor lasergamen. Dit lasergame spel wordt gespeeld met behulp van een arduino due. Deze due beschikt over een ARM-processor met 1 kern. Dit brengt een problemen met zich mee hoe kunnen twee processen bijvoorbeeld tegelijk runnen op maar één processorkern.

Om dit soort problemen te kunnen oplossen gaan wij het arduino RTOS gebruiken. Hierover kunt u later meer vinden in ons onderzoeksrapport. Het arduino RTOS is echter niet de enige soort RTOS die hiervoor gebruikt kan worden. In dit onderzoeksrapport gaan wij de geschiktheid van de verschillende soorten RTOS-systemen bekijken. Hiermee willen we bereiken dat we naast het arduino RTOS ook nog een andere RTOS hebben die we voor het project zouden kunnen gebruiken.

De opbouw van dit rapport ziet er als volgt uit: Na de inleiding wordt de probleemstelling en de daarop gebaseerde deelvragen besproken. Daarnaast gaan we het hebben over welke methodes in het onderzoek gebruikt zijn. Hierna gaan we eerst goed kijken naar de arduino RTOS zelf. Ook gaan we het hebben over de verschillende soorten RTOS-systemen die er zijn en wat deze voor overeenkomsten en verschillen hebben. Daarnaast hebben we het over hoe we die verschillen eventueel kunnen overbruggen met oplossingen die binnen de RTOS mogelijk zijn. Tot slot geven we de conclusie van het onderzoeksrapport en een aanbeveling.

# Probleemstelling en onderzoeksvragen

Het arduino RTOS heeft een aantal functionaliteiten die wij in ons concurrency diagram gebruikt hebben. Deze functionaliteiten zijn: een channel, een clock, een pool, een flag, een mailbox en een mutex. Voor dit onderzoek gaan we vooral gebruik maken van de arduino RTOS-documentatie, daarnaast zullen in onze bronvermelding alle webpagina’s staat die gebruikt worden om de voor de andere RTOS-systemen. De functionaliteiten van de arduino RTOS moeten deels terugkomen in de andere RTOS. Aan de hand van deze informatie hebben we de volgende hoofdvraag opgesteld:

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

Hierna hebben we de volgende deelvragen opgesteld:

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

Hierbij hebben gaan we vooral kijken naar wat het arduino RTOS voor eigenschappen heeft en hoe deze worden gebruikt.

* *Welk open source RTOS-en zijn beschikbaar?*

Aan deze deelvraag hebben we nog wat extra eisen verbonden om de RTOS-systemen geschikter te maken. Deze staan hieronder weergeven:

* *Het RTOS moet beschikbaar zijn voor het ARM-platform*
* *Het RTOS moet C en C++ ondersteunen*
* *Het RTOS moet binnen het afgelopen jaar een update hebben gekregen*
* *Het RTOS moet beschikken over documentatie waarin de geboden functionaliteit is beschreven*
* *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?*

Bij deze deelvraag gaan we de informatie van de vorige deelvragen gebruiken en de verschillende RTOS-systemen met elkaar vergelijken

* *Welke mechanismen van het Arduino RTOS worden niet ondersteund door de beschikbare RTOS-en?*

Bij deze deelvraag zijn we vooral bezig met het vinden van verschillen in de RTOS-systemen

* *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?*

Bij deze deelvraag gaan we opzoek naar oplossingen voor de gevonden verschillen.

# Onderzoeksmethoden

Wij gaan ons onderzoekrapport uitvoeren door veel research te doen. We gaan dus op zoek naar informatie, en leggen geen interviews of enquêtes af. We gaan dus vooral literatuurstudie die en vergelijken bestaande oplossingen

Onze onderzoeksmethode voor literatuurstudie zal vooral bestaan uit het doorlezen van doxygen documentatie voor het arduino RTOS en de andere RTOS-systemen. Op het internet kunnen heel erg veel oplossingen gevonden worden voor de problemen die we ongetwijfeld tegen zullen komen. Hierbij passen wij onze tweede onderzoeksmethode toe, vergelijken bestaande oplossingen. Deze methode wordt ook vooral toegepast op de laatste deelvraag. Hierbij gaan we kijken naar de oplossingen voor ontbrekende delen in de andere RTOS-systemen

**Hier deelvragen plaatsen**

**Conclusies en aanbeveling**

De hoofdvraag van dit onderzoeksrapport luide:

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

Door de verschillende deelvragen te beantwoorden zijn wij tot de conclusie gekomen dat het ….. RTOS-systeem hier het meest geschikt voor is.

**Hier uitleggen op welke deelvragen dit is gebaseerd.**

**Aanbeveling**

Op grond van deze conclusie doen we een aantal aanbevelingen over het gebruik van dit RTOS-systeem bij dit specifiek project:

**Aanbevelingen hier**