RTOS research

Robert Bezem, Luuk Steeman,
Ricardo Bouwman, Jeroen van Hattem

Oktober 2016

Research naar verschillende RTOS-en en vergelijking met het Arduino RTOS

Inhoudsopgave

Deel I
Introductie

Door de Hoge School Utrecht zijn wij gevraagd om onderzoek te doen naar verschillende Real-Time Operating Systems. In dit onderzoeksrapport beschrijven wij onze bevindingen over de verschillende RTOS-en die beschikbaar zijn voor ARM-processoren.

1 Hoofdvraag

Met behulp van welk open source real-time 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?

2 Deelvragen

- Wat zijn de kenmerkende eigenschappen van tasks en de concurrency mechanismen van het Arduino RTOS?
- Welke open source RTOS-en zijn beschikbaar?
- Welke 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?
- Welke mechanismen van het Arduino RTOS worden niet ondersteund door de beschikbare RTOS-en?
- 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?

Deel II
Onderzoek

3 Kenmerkende eigenschappen Arduino RTOS

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

3.1 Onderzoeksmethode

Om de eigenschappen van het Arduino RTOS te onderzoeken is gekeken naar de Doxygen documentatie en de source code zelf. De Doxygen documentatie is te genereren door het volgende commando uit te voeren in de bestandsmap waar de sourcecode zich bevindt

```
doxygen rtos.hpp
```

de documentatie is daarna te vinden in *html/index.html* in HTML voormaat.

3.2 Onderzoeksresultaten

Het ArduinoRTOS is een cooperative RTOS geschreven in C++ en assembly. Er worden verschillende concurrecy mechanismen geboden door dit RTOS, bij namen:

- Pool
- Mutex
- Channel
- Mailbox
- Timer
- Clock
- Flag

4 Beschikbare open source RTOS-en

Welke open source RTOS-en zijn beschikbaar.

4.1 Onderzoeksmethode

Omdat het aantal bestaande open source RTOS-en ontzettend groot is zijn er een aantal initiele criteria opgesteld om een eerste selectie te maken.

- 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

Omdat er na deze eerste selectie nog te veel verschillende RTOS-en over blijven zijn er 8 willekeurige RTOS-en uitgezocht voor nader onderzoek. Wij hebben de volgende twee sites gebruikt om opensource RTOS-en te vinden:

- Comparison of real-time operating systems (et al., 2016)
- osRTOS (OSRTOS, 2016)

4.2 Onderzoeksresultaten

Na de selectie gemaakt te hebben is er gekozen om de volgende RTOS-en te onderzoeken.

- StratifyOS (tyler gilbert, 2016)
- NuttX (patacongo, 2016)
- Zephyr (Zephyr, 2016)
- Mark3OS (Mark3OS, 2016)
- Free-RTOS(free rtos, 2016)
- Distortos (Distoros, 2016)
- RTEMS (RTEMS, 2016)
- mbedOS (ARM Ltd, 2016)

5 Vergelijking beschikbare mechanismen

In dit hoofdstuk worden twee deelvragen tegelijk beantwoord: Welke 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? en Welke mechanismen van het Arduino RTOS worden niet ondersteund door de beschikbare RTOS-en?

5.1 Ondzoeksmethode

Om de verschillende mechanismen ondersteund door de RTOS-en te vinden is er gekeken naar de documentatie van deze RTOS-en.

5.2 Onderzoeksresultaten

Naam RTOS	Pool	Mutex	Channel	Mailbox	Timer	Clock	Flag
StratifyOS		•	•		•	•	•
NuttX		•	•		•	•	•
Zephyr	•	•	•	•	•	•	•
Mark3OS	•	•	•	•	•	•	•
Free-RTOS		•	•		•	•	•
DistortOS		•	•		•	•	•
RTEMS(OAR Corporation, 2008)		•	•	•	•	•	•
mbedOS(Ltd, 2016)	•	•	•	•	•	•	•

6 Vervanging niet ondersteunde mechanismen

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?

6.1 Ondzoeksmethode

Hiervoor is de implementatie van niet beschikbare mechanismen in het RTOS bestudeerd.

6.2 Onderzoeksresultaten

6.2.1 Pool

Een pool kan worden geïmiteerd door middel van een globale variabele en een mutex. Een pool kan worden gezien als een variabele die kan gelezen en overschreven kan worden door meerdere tasks, maar dit niet tegelijk. Door middel van een globale variabele en een mutex kan dit ook. Wanneer een task de variabele wilt schrijven, locked hij de mutex, wanneer hij klaar is, geeft hij hem vrij. Hetzelfde geldt voor lezen. Dit is zo zodat de variabel niet kan worden veranderd wanneer hij wordt gelezen.

6.2.2 Channel

Een channel kan geïmplementeerd worden door middel van een flag, mutex en een array. Deze C++ code geeft een opzet aan hoe een channel geïmplementeerd kan worden

```
template <typename T, int L>
class Channel
{
    T queue[L];
    int length;
    flag theFlag;

    void addToQueue(T elem)
    {
        mutex.lock();
        queue.add(elem);
        length++;
        flag.signal();
        mutex.unlock();
    }

    T wait()
    {
        theFlag.wait();
        T returnVal = queue[0];
        for (int i = 0; i < length; i++)
        {
            queue[i] = queue[i + 1];
        }
        len--;
        return returnVal;
    }
}
```

6.2.3 Mailbox

Een mailbox kan worden nagebootst door middel van channel en een binary semaphore. Een mailbox houdt in dat er data in kan worden gezet en kan worden uitgelezen. Een binary semaphore kan worden gezien als een mutex met een counter. Deze counter wordt op een nummer gezet, de task die iets wilt doen, moet de semaphore daarom vragen. Elke keer dat er iets aan de semaphore wordt gevraagd, wordt deze counter met 1 verlaagd. Dit kan als mailbox worden gebruikt door de channel te "vergrendelen" wanneer erin is geschreven. De lezer zal dan eerst meerdere keren moeten proberen het te kunnen lezen. Op die manier wordt het systeem, net als bij een mailbox, geblokkeerd voor een bepaalde periode.

Deel III
Conclusie & aanbeveling

7 Conlclusie

Van alle RTOS-en waren er drie die alle concurrency mechanismes die we zochten hadden.

- Zephyr
- Mark3OS
- mbedOS

Om een volledige conclusie te kunnen trekken, kan er gekeken worden naar de documentatie en hoe ingewikkeld het is om het RTOS te implementeren.
Mark3OS en mbedOS zijn zeer goed gedocumenteerd. Bij alles staat heel duidelijk wat het doet en hoe het toegepast moet worden. Ze hebben ook een uitgebreid "getting started"deel om programmeurs op weg te helpen. In de documentatie op de website van mbedOS staan ook afbeeldingen om concurrency mechanismes uit te leggen, maar het is soms ingewikkeld om de documentatie van een specifiek deel te vinden.
Mark3OS heeft geen plaatjes, maar de documentatie is beter gestructureerd en het is dus makkelijker om van een specifiek onderdeel te vinden hoe het werkt en hoe je het moet toepassen.

8 Aanbeveling

Wij bevelen Mark3OS aan als het meest geschikte RTOS. Het bevat alle benodigde concurrency mechanismes, het is goed gedocumenteerd en goed begrijpbaar voor programmeurs die nog niet eerder met RTOS-en hebben gewerkt.

Literatuur

ARM Ltd. (2016). *Home — mbed.* Verkregen van https://www.mbed.com/en/

Distoros. (2016). *Api distortos documentation.* Verkregen van http://distortos.org/api-reference/

et al., E. (2016, 23 oktober). *Comparison of realtime operating systems.* Verkregen van https://en.wikipedia.org/wiki/Comparison_of_real\penalty\@M\hskip\z@skip\discretionary{-}{}{}\penalty\@M\hskip\z@skiptime_operating_systems

free rtos. (2016). *Homepage free-rtos.* Verkregen van http://www.freertos.org/

Ltd, A. (2016). *Rtos - mbed os api references.* Verkregen van https://docs.mbed.com/docs/mbed-os-api-reference/en/5.2/APIs/tasks/rtos/

Mark3OS. (2016, 11 maart). *Mark3 realtime kernel class list.* Verkregen van http://mark3os.com/docs/html/annotated.html

OAR Corporation. (2008). *Rtems 4.9.6 on-line library.* Verkregen van https://docs.rtems.org/releases/rtemsdocs-4.9.6/share/rtems/html/

OSRTOS. (2016, 09 oktober). *Open source rtos.* Verkregen van http://www.osrtos.com/

patacongo. (2016, 25 juli). *Nuttx real-time operating system.* Verkregen van http://nuttx.org/

RTEMS. (2016, juni). *Rtems real time operating system (rtos) — real-time and real free rtos.* Verkregen van https://www.rtems.org/

tyler gilbert. (2016, 29 oktober). *Stratifylabs/stratifyos: A powerful embedded rtos for the arm cortex m microcontrollers.* Verkregen van https://github.com/StratifyLabs/StratifyOS

Zephyr. (2016, 31 oktober). *Microkernel synchronization services.* Verkregen van https://www.zephyrproject.org/doc/kernel/microkernel/microkernel_synchronization.html