Practicum 4

De scheduler – deel 2

In dit practicum zullen we de scheduler van ons rudimentair besturingssysteem aanpassen zodat meerdere taken kunnen worden uitgevoerd op de computer. In practicum 3 was het slechts mogelijk 1 enkele taak uit te voeren, namelijk *spiraal*. Vanzelfsprekend kunnen deze taken in een uniprocessorsysteem niet tegelijk worden uitgevoerd, maar door het snel wisselen tussen taken, krijgt de gebruiker toch de indruk dat de taken simultaan lopen.

4.1 Indienen

Dit practicum wordt gequoteerd. Je moet je code indienen voor (**woensdag 9 april 2014 om 22:00:00**) via `https://indianio.ugent.be`. Let op! We verwachten per groep één enkel zip bestand waarin de volgende bestanden zijn opgenomen.

- `practicum4_vraag1.asm`
- `practicum4_vraag3.asm`
- `practicum4_vraag4.asm`
- `practicum4_vraag5.asm`

Zoals steeds gelden de algemene richtlijnen voor het indienen van verslagen.

We vertrekken van een correcte implementatie van labo 3, die we reeds voor jullie in de directory *pract04* hebben gezet.

4.2 Opgaven

1. Nu we taken kunnen aanmaken, willen we ook taken kunnen beëindigen, zodat deze niet langer gescheduled worden. Implementeer hiertoe de routine `termineertaak`.

   Deze routine moet het element dat verwijst naar de stapel van de taak behorende bij de `Huidige_Taak` in de takenlijst op 0 zetten, een andere taak zoeken die mag worden uitgevoerd worden en die dan instellen als huidige taak. Nu kan je naar believen taken opstarten en laten termineren.

   Sla je bestand op als `practicum4_vraag1.asm`.

2. Op dit punt kunnen we zonder problemen de oneindige lus in de hoofdtaak wegwerken door `termineertaak` op te roepen in plaats van de oneindige lus uit te voeren. Als je de vorige stappen correct hebt uitgevoerd, zal dit werken.

   Je kan dit eenvoudig verifiëren door naar de uitvoer van de informatietaak te kijken: het slot van de main-functie waarbij voorheen een 'A' werd geprint, zal nu een 'T' bevatten.

3. We willen er nu voor zorgen dat de scheduler aangeeft wanneer er geen taken kunnen worden uitgevoerd. Zorg ervoor dat in dat geval een boodschap geprint wordt, bijvoorbeeld `GEEN TAKEN`. Om dit te testen moet je de spiraal termineren na X ticks, bijvoorbeeld X=5000. Je taken termineer je door in de 'herstart' functionaliteit van spiraal op de aangeduide plaats te controlleren of Huidige_Tick deze vooraf bepaalde waarde overschrijft, en zo nodig termineertaak op te roepen. (Uiteraard ga je dan voor deze opgave dan ofwel de InfoTaak niet opstarten, of ga je ze ook moeten termineren na X ticks.)

   Sla je bestand op als `practicum4_vraag3.asm`.

4. Op dit moment zal de schedulerhandler nog steeds in een oneindige lus vastzitten. In plaats van deze conditie te detecteren en te melden, zouden we deze ook kunnen opvangen.

   Hiertoe zullen we een zogenaamde idle taak toevoegen aan de takenlijst. Dit is een taak die bestaat uit een oneindige lus, en deze taak zal énkel worden uitgevoerd indien de schedulerhandler geen andere taken meer kan vinden om uit te voeren. Hierbij hoef je dus niét meer `GEEN TAKEN` te printen. (Maar, om te verifiëren of dit werkt, zal je net zoals in de vorige opgave een taak termineren na X=5000 ticks.)

   Om dit te implementeren, zal je de functie `creeer_idle_taak` moeten implementeren en aanroepen om de idle taak in het `idle_taak_slot` te installeren, en zal je de schedulerhandler moeten aanpassen zodat deze, ten gepasten tijde, de idle taak activeert. Als er nadien terug een andere taak wordt gecreëerd, moet deze opnieuw worden uitgevoerd in plaats van de idle taak.

   Sla je bestand op als `practicum4_vraag4.asm`.

5. Een groot voordeel van de implementatie uit opgave 4 in vergelijking met opgave 3 is dat de schedulerhandler nog correct blijft werken, ook al zijn er op een bepaald punt geen taken meer uit te voeren. Dit is bijzonder handig bij het implementeren van een sleep- functie: een taak kan vragen om een aantal klokticks niet uit te voeren; als alle taken dit tegelijk doen, zal er tijdens die kloktick geen taak meer te schedulen zijn, maar omdat de schedulerhandler nog blijft werken, zullen de taken na een tijd terug uitgevoerd kunnen worden.

   We gaan dit gebruiken om de spiralen iets minder epileptisch snel op het scherm te laten flikkeren. Wij hebben voor jullie reeds een `sleep`-functie geschreven. Roep deze nu aan vanuit de functie `ShortDelay` met redelijke argumenten, zorg er voor dat de taken terug oneindig lang blijven lopen (in plaats van getermineerd te worden na 5000 ticks), en check dat je aangepaste schedulerhandler inderdaad correct werkt in dit geval, en de spiralen trager getekend worden.

   Sla je bestand op als `practicum4_vraag5.asm`.

Succes!