

# ICT Infrastructuren (I00038), herfstsemester 2006/2007

Eric D. Schabell  
Instituut voor Informatica en Informatiekunde (ICIS)  
Radboud Universiteit Nijmegen  
<http://www.schabell.com/icti>  
<mailto:erics@cs.ru.nl>

September 28, 2006

## Opdracht B

In deze opdracht schrijf je de multi-threaded controle software voor een een *wafer scanner*. Dit is een apparaat dat chipdesigns op plakken silicium, zogenaamde wafers, projecteert. Een chemisch proces zorgt er vervolgens voor dat de chip in het silicium wordt gevormd. Op <http://www.asml.com/> kun je meer vinden over deze machines die functioneren op de grenzen van wat mogelijk is. Het is aardig om op te merken dat deze opdracht aan een realistische case-study is ontleend.

### De Inleiding

De wafers komen onze wafer scanner binnen via twee *locks* en komen ook via deze locks naar buiten. Elk lock kan maximaal 1 wafer bevatten. De zogenaamde *track robot* staat buiten de machine en zorgt voor de aanvoer van verse wafers en voor de afvoer van belichte wafers. De track robot wordt bestuurd met twee commando's, geparametriseerd door het lock nummer:

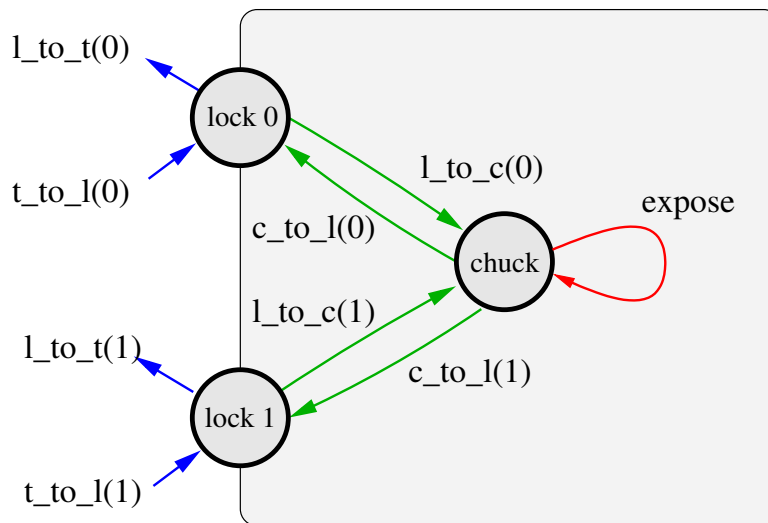
- `track_to_lock(i)` ( $i \in \{0, 1\}$ ) verplaatst een wafer van de track naar lock  $i$ .
- `lock_to_track(i)` ( $i \in \{0, 1\}$ ) verplaatst een wafer van lock  $i$  naar de track.

Binnen in de machine staat de *interne robot* die wafers van de twee locks naar de zogenaamde *chuck* verplaatst en ook weer terug. Ook de chuck kan maximaal 1 wafer bevatten. De interne robot kent de volgende besturingsfuncties:

- `lock_to_chuck(i)` ( $i \in \{0, 1\}$ ) verplaatst een wafer van lock  $i$  naar de chuck.
- `chuck_to_lock(i)` ( $i \in \{0, 1\}$ ) verplaatst een wafer van de chuck naar lock  $i$ .

Op de chuck vindt de eigenlijke belichtings (*exposure*) operatie plaats. Deze operatie wordt uitgevoerd door de volgende functie:

- `expose_operation()` belicht een verse wafer.



Figuur 1: Het schema met de materiaalpaden en de operaties in onze wafer scanner. De track robot voert de blauwe operaties uit, de interne robot voert de groene operaties uit, en de belichtingseenheid voert de rode operatie uit.

De functie van de machine is om zo snel mogelijk wafers precies 1 maal te belichten. Het een en ander is schematisch weergegeven in Figuur 1.

De controle software van de machine kan logischerwijze in drie stukken worden onderverdeeld: de besturing van de track robot, de besturing van de interne robot, en de besturing van de expose operatie. Elk stuk zal worden geïmplementeerd door een aparte thread binnen het proces dat de machine bestuurt. Nu zijn er natuurlijk een aantal dingen waarop gelet moet worden:

- Deadlock van de machine is zeer ongewenst omdat de hele productie dan moet worden stilgelegd. Dat kost een hele hoop geld. Een gerelateerd issue is de zogenaamde “livelock”. Dat is een situatie waarin er wel dingen gebeuren, maar die geen netto effect hebben. Je kan denken aan het continu verplaatsen van een wafer van een lock naar de chuck en terug. Het is duidelijk dat zulke situaties ook voorkomen moeten worden.
- Throughput is zeer belangrijk: het is een van de “key performance criteria” van zulke machines. Daarom is het zaak om zoveel mogelijk parallellisme in te bouwen. Merk op dat de deadlock en livelock issues uit het vorige item en throughput optimalisatie twee belangen zijn die elkaar in de weg lopen: Je kunt makkelijk deadlock voorkomen door een hele simpele besturingsstrategie te implementeren die op elk moment maar 1 wafer in de machine toelaat. Zo’n strategie heeft echter duidelijk een suboptimale throughput en is daarom niet gewenst.
- Er zijn natuurlijk een aantal operaties die niet tegelijk kunnen gebeuren omdat er dan delen van de machine fysiek met elkaar in botsing komen:
  - Als de trackrobot met lock  $i$  bezig is, dan kan de interne robot niets met lock  $i$  doen, en vice versa.

- Als de expose operatie bezig is, dan kan de interne robot niets met de chuck doen, en vice versa.

## De Opdracht

Download <http://www.schabell.com/icti/exercise-B/practicalB.tar> en pak het uit. Je ziet 2 files: `opdracht_B.c` en `Makefile`. Met het welbekende `make` commando kun je het C file compileren tot een executable `opdracht_B`. Het C file is een raamwerk voor de besturingssoftware van de machine. Het is nu jouw taak om de software af te maken: implementeer de functies `track_robot`, `internal_robot` en als laatste `expose_operation` die allen als threads worden opgestart in de functie `main`. Tips:

- Elke thread heeft een pointer naar dezelfde `state_t` structuur die de huidige toestand van de machine moet bevatten, en die door de threads dus gebruikt kan worden om de te volgen actie te bepalen. Deze `state_t` structuur bevat informatie over welke wafers waar liggen (zie de C code voor meer details). Zorg ervoor dat de huidige toestand van de machine altijd actueel is.
- Je kunt gebruik maken van een zelf te bepalen aantal semaforen (zie de C code voor de details; merk op dat semaforen in de `main` functie dienen te worden geïnitieerd).
- Voorkom deadlock en livelock, en natuurlijk ook incorrecte parallele besturingsacties. Belicht wafers precies 1 keer, en laat alleen belichte wafers uit de machine komen.
- Zorg ervoor dat de machine ook nog een goede throughput heeft. Dus: triviale volledig sequentiele oplossingen zijn niet voldoende.
- Documenteer je oplossing zorgvuldig. Een goede uitleg over waarom jouw oplossing correct is, is even belangrijk als de implementatie. Behandel ook in ieder geval de algemene gedachte achter je oplossing: maak je gebruik van preventie, voorkoming of detectie van deadlock, en waarom?

Inleveren op maandag 30 oktober, voor college begint: het C programma met goede documentatie van de oplossing (als commentaar in de C code).