1 Inleiding

Een toestandsautomaat vormt een systeembeschrijving waarbij een systeem telkens vanuit een toestand naar een andere toestand kan gaan bij het optreden van een bepaalde invoergebeurtenis.

In het meest eenvoudige geval betekent het dat de verwerkingseenheid van het systeem niets doet tenzij er een invoergebeurtenis optreedt.

Als voorbeeld wordt hiervoor een lamp genomen, die met behulp van een terugverende drukknop geschakeld kan worden, waarbij het programma denkbeeldig de knop vasthoudt.

Als gebeurtenissen zien we hierbij:

in de knop wordt ingedrukt tegen zijn veerdruk in

los de knop wordt losgelaten en veert terug uit

De onderscheiden stabiele toestanden zijn:

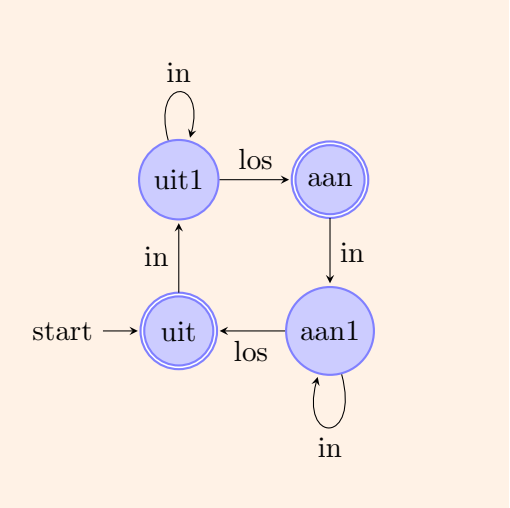
uit de lamp is uit, ook als begintoestand gekozen

aan de lamp is aan

We willen dat de lamp 'natuurlijk' reageert, dus aan op indrukken als de lamp uit is, en uit op loslaten als de lamp aan is. Bovendien willen we een aantal milliseconden contactdender van de drukknop onderdrukken. Daarvoor is het handig om 2 overgangstoestanden toe te voegen:

uit1 de lamp was uit en de knop ingedrukt

aan1 de lamp was aan en de knop ingedrukt



Het voorbeeld is te programmeren in een 4×2 beslismatrix met rijen voor de toestanden en kolommen voor de optredende invoer. De matrix wordt ingevuld met een minteken voor een combinatie waar niets wijzigt, of de nieuwe toestand voor de combinatie, eventueel voorafgegaan door, schuingedrukt, de te nemen actie op de lampuitgang, en een komma als scheidingsteken:

	in	los
uit	*ga aan*, uit1	-
uit1	-	aan
aan	aan1	-
aan1	-	*ga uit*, uit

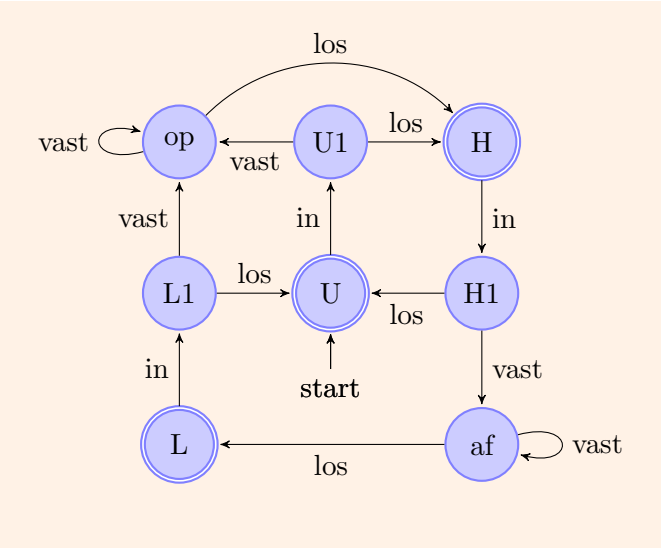
2 dimmerautomaat

Als bij de inleiding willen we dat een lamp die uit is meteen inschakelt bij indrukken, en een lamp die aan is pas uitschakelt bij loslaten.

Het toestandsdiagram uit de inleiding wordt uitgebreid met vast ingedrukt houden van de knop, waarbij op- of afdimmen plaatsvindt. Daartoe wordt een nieuwe gebeurtenis (kolom) **vast** in de tabel opgenomen, die regelmatig herhaalt bij ingedrukt houden. Op deze gebeurtenis zijn twee toestanden nodig, **op** en **af**, waarin de helderheid van de lamp wordt bijgeregeld.

Verder willen we een geheugen hebben om net voor het uitschakelen de laatst gebruikte helderheid in te bewaren, en net voor het inschakelen deze weer uit op te halen, zodat bij voorbeeld een sfeerlamp op een gekozen helderheid wordt ingeschakeld.

We ontdekken dat daarvoor 2 stabiele aan-toestanden nodig zijn. De bestaande aan en aan1 worden hernoemd naar H en H1 voor hoge helderheid, en de nieuwe noemen we L voor lage helderheid, met L1 om ook vanaf L met een korte druk de lamp uit te schakelen. Na hernoemen van uit en uit1 naar U en U1 ontstaat een elegant diagram met diagonalen voor acties/stabiele toestanden en zijden voor de overgangstoestanden:



De bijbehorende matrix met mogelijke implementatiefuncties ernaast wordt:

	los	in	vast
U	-	f1()	-
U1	h()	-	f2()
op	h()	-	f2()
H	-	h1()	-
H1	f4()	-	f3()
af	l()	-	f3()
L	-	l1()	-
L1	f4()	-	f2()

```
f1() {return lamp=bewaard, U1;}
f2() {if(++lamp>max) lamp=max; return op;}
f3() {if(--lamp<min) lamp=min; return af;}
f4() {return bewaard=lamp,lamp=0, U;}
h() {return H;}
h1() {return H1;}
l() {return L;}
l1() {return L1;}
```

In een echt project kan ik bovenstaande optimaliseren en met meer functionaliteit uitbreiden.