

Dit document is een template voor een 'Design Description' van een project. In de blauwe verborgen tekst is uitleg gegeven over de betreffende hoofdstukken.

Deze verborgen teksten kunnen zichtbaar gemaakt worden met de "Weergeven/verbergen ¶" knop in de werkbalk. De tekst kan ook weergegeven worden via Extra, Opties..., tab Weergave, waar "Verborgen tekst" kan worden aangevinkt. Bij het afdrukken kun je met het aanvinken van "Verborgen tekst" onder afdrukopties aangeven of de verborgen tekst moet worden afgedrukt of niet.

Je kunt dit template gebruiken door ook deze blauwe tekst te verbergen (kan door tekst te selecteren, dan via Opmaak, Lettertype en daar het veld "Verborgen tekst" aanvinken) of te verwijderen.

## **Design Description**

**voor**

# **Hobbybrouwen PID Regelaar**

**Opgesteld door** Emile van de Logt

**Projectleider**

**Projectleden**

**Datum van uitgifte** Juni 2007

**Referentie** Ontwerp\_PID\_regelaar\_v025.doc

**© 2009 Emile van de Logt**

## Versiebeheer

<b>Versiehistorie</b>			
<b>Versie</b>	<b>Datum</b>	<b>Wijzigingen</b>	<b>Auteur</b>
0.1	17-11-2006	Eerste versie: H1 en H2 ingevuld	E vd Logt
0.2	10-06-2007	Diverse aanvullingen, geschikt gemaakt voor publicatie	E vd Logt
0.25	29-04-2009	Diverse aanvullingen, aangepast aan v0.4 van specificatie. Gepubliceerd op website	E vd Logt

## Inhoudsopgave

1	Introductie .....	4
1.1	Doel .....	4
1.2	Document conventies.....	4
1.3	Doelgroep en suggesties t.b.v. het lezen.....	4
1.4	Scope van het project .....	4
1.5	Referenties.....	4
2	Architectuur Ontwerp .....	5
2.1	Het contextdiagram .....	5
2.2	Ontwerpbesluiten .....	7
2.3	Product / Systeem Componenten.....	7
2.4	Beschrijving van de software deelcomponenten .....	8
2.4.1	Deelcomponent 1: Verwerk Sensor signalen.....	8
2.4.2	Deelcomponent 2: Verwerk Inputs.....	8
2.4.3	Deelcomponent 3: Bepaal Referentie Temp.....	8
2.4.4	Deelcomponent 4: Regel Temperatuur.....	8
2.4.5	Deelcomponent 5: Genereer Outputs .....	8
2.4.6	Deelcomponent 6: Display en Log Signalen.....	8
2.5	Beschrijving van de hardware deelcomponenten .....	8
2.6	Beschrijving van de Interfaces .....	9
3	Detail Ontwerp .....	10
3.1	Deelcomponent 1: Verwerk Sensor signalen.....	10
3.1.1	Deelcomponent 1.1: Verwerk Thermokoppel .....	10
3.1.2	Deelcomponent 1.2: Verwerk PT100 .....	11
3.1.3	Deelcomponent 1.3: Verwerk 1Wire .....	12
3.1.4	Deelcomponent 1.4: Verwerk I2C.....	12
3.1.5	Deelcomponent 1.5: Verwerk Temperaturen.....	13
3.2	Deelcomponent 2: Verwerk Inputs.....	13
3.3	Deelcomponent 3: Bepaal Referentie Temp.....	13
3.4	Deelcomponent 4: Regel Temperatuur.....	14
3.5	Deelcomponent 5: Genereer Outputs .....	14
3.6	Deelcomponent 6: Display en Log Signalen.....	14
4	Requirements Traceability .....	15
Appendix 1.	AT91SAM7S64 PIO Port Mapping .....	16
Appendix 2.	Parameters and Variables List.....	17
Appendix 3.	How to operate the Menu and the Display.....	19
Appendix 4.	RS232 Serial port 0: available commands.....	21

## 1 **Introductie**

### 1.1 **Doel**

De doelstelling van dit project is om een PID regelaar te ontwerpen die specifieke taken kan uitvoeren voor een hobbybrouwer. Na discussie met deze hobbybrouwers worden een aantal scenario's gedefinieerd, waarvoor deze regelaar ingezet kan worden (zie het specificatie document voor een uitwerking hiervan).

### 1.2 **Document conventies**

Dit document beschrijft een ontwerp voor een PID regelaar. Zo'n PID regelaar bestaat uit de nodige elektronica. Om dit goed te kunnen ontwerpen zijn een aantal specifieke technieken gebruikt. De hier gebruikte ontwerpbenadering is gebaseerd op functionele decompositie, de toegepaste methodiek is die van Hatley & Pirbhai.

Het ontwerpmodel, waarvan de plaatjes en figuren ook in dit document staan, is gerealiseerd m.b.v. een CASE-tool (Computer Aided Software Engineering), namelijk de tool AxiomSys, die de methode van Hatley & Pirbhai volledig ondersteunt.

Voor specifieke vragen over deze methode wordt verwezen naar het boek van Hatley & Pirbhai (Strategies for Real-Time System Specification, Dorset House Publishing, ISBN 0-932633-11-0). De daar beschreven conventies worden in dit document integraal toegepast.

### 1.3 **Doelgroep en suggesties t.b.v. het lezen**

De belangrijkste doelgroepen zijn de volgende:

- Ontwerpers: die willen meedenken met het ontwerp en hier commentaar op willen leveren. Zij zijn goed bekend met de hiervoor genoemde ontwerpmethodiek.
- Hobbybrouwers: die willen een algemeen beeld krijgen van de opzet van zo'n regelaar. Voor hen zijn m.n. dit hoofdstuk en hoofdstuk 2 (Architectuurontwerp) interessant. De overige hoofdstukken kunnen teveel in detail gaan.

### 1.4 **Scope van het project**

De scope van dit project omvat de specificatie, het ontwerp, de bouw en test van de complete PID regelaar. Ook een gebruikershandleiding hoort hierbij.

Verder zal er een PC applicatie gemaakt worden, waarmee makkelijk parameters van de PID regelaar uitgelezen en ingesteld kunnen worden.

### 1.5 **Referenties**

De belangrijkste referentie is het specificatiedocument, Specificatie\_PID\_regelaar\_v04.doc. Deze dient als basis voor dit project.

Daarnaast wordt op het Hobbybrouwen forum actief over dit project gediscussieerd, zie <http://www.hobbybrouwen.nl/forum/index.php/topic,4988.0.html>.

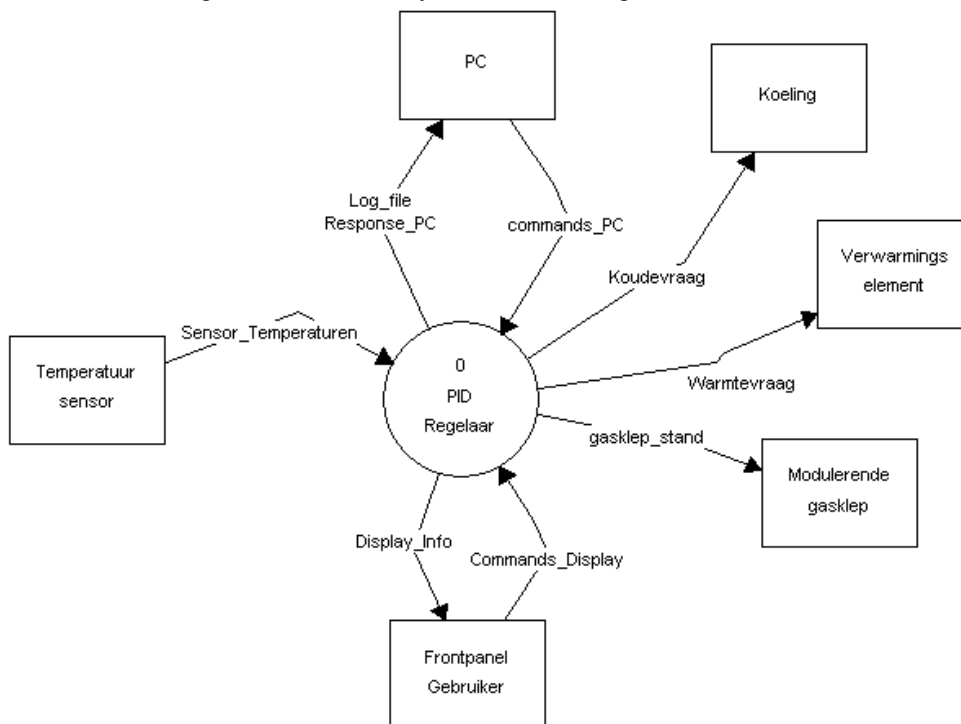
## 2 Architectuur Ontwerp

Het architectuurontwerp is het eerste niveau in het totale ontwerp. Dit architectuurontwerp is verdeeld in de volgende paragrafen:

- Het Context Diagram
- Ontwerpbesluiten
- Product / Stelselcomponenten
- Beschrijving van de software deelcomponenten
- Beschrijving van de hardware deelcomponenten
- Beschrijving van de interfaces

### 2.1 Het contextdiagram

Een eerste afbakening van het stelsel dient om de context te definiëren. Hierin wordt aangegeven wat wel bij het stelsel hoort en wat er niet bij hoort. Hieronder staat het Context Diagram van de Hobbybrouwen PID regelaar:



**Figuur 1: Context Diagram van de Hobbybrouwen PID regelaar**

De bol in het midden is het te realiseren stelsel. Deze communiceert met terminators (de buitenwereld). Dit zijn de volgende terminators:

- **PC**: Via een PC kan een gebruiker communiceren met het stelsel. Een PC hoeft niet aangesloten te zijn voor een correcte werking van het stelsel. De PC dient puur om op een handige manier instellingen te kunnen verzorgen.
- **Koeling**: Om bij het climate-control scenario (zie specificatie document) een klimaatkast te kunnen koelen, dient de motor / compressor van een koelkast / diepvries aangestuurd te kunnen worden. Er wordt van uitgegaan dat het koelvermogen maximaal 1000 Watt is bij 230 V (zie requirement R10b)
- **Verwarmingselement**: Het verwarmingselement dat in de warmwaterketel ligt. Er wordt van uitgegaan dat het verwarmingselement maximaal 3000 Watt is bij 230 Volt AC ( zie requirement R10a).
- **Modulerende gasklep**: Een modulerende gasklep kan gebruikt worden om een gasbrander aan te sturen. Zo'n modulerende gasklep verwacht een pulsbreedte gemoduleerd signaal (28 Volt DC), waarvan de pulsbreedte gevarieerd wordt tussen 0% en 100%, afhankelijk van de warmtevraag (zie requirement R11).

- Frontpanel gebruiker: Dit is het bedieningspaneel van de PID regelaar. Hiermee kan de gebruiker een aantal instellingen verzorgen en enkele waarden aflezen van het display.
- Temperatuursensor: Een temperatuursensor kan volgens de specificatie bestaan uit de volgende onderdelen (zie requirement R8):
  1. Een thermokoppel: maximaal 1
  2. Een Pt100 element: maximaal 1
  3. Een One-Wire digitale temperatuursensor: maximaal 2
  4. Een I<sup>2</sup>C digitale temperatuursensor: maximaal 2Alle sensoren worden ondersteund. Er kunnen maximaal 2 sensoren tegelijkertijd aangesloten zijn (zie requirement R7).

Tussen de terminators (de buitenwereld) en de PID regelaar lopen signalen (de "data-flows"). Deze data-flows zijn:

- Sensor Temperaturen: De temperaturen van de verschillende sensoren. In totaal kunnen 6 sensoren aangesloten worden, waarvan 2 op hetzelfde moment.
- Log file: De log-file kan naar de PC gestuurd worden, indien daar vanuit de PC om gevraagd wordt. De log\_file bevat de volgende informatie: act\_temp1, act\_emp2, pid\_output, ref\_temp en sys\_mode  
Deze informatie wordt iedere 5 seconden geregistreerd en opgeslagen.
- Response PC: De respons van de PID-regelaar op een commando vanuit de PC. Meestal betreft dit de gevraagde respons of een bevestiging van de verwerking van commando vanuit de PC.
- Commands PC: Vanuit de PC kunnen een aantal commando's verstuurd worden naar de PID-regelaar. Dit zijn de volgende commando's (zie Appendix 4 voor details):
  - log <on, off>: zet logging naar de PC aan of uit
  - pid <on, off>: zet de PID regelaar aan of uit
  - get vaxx: laat de waarde zien van variabele xx
  - set vaxx yyyy: fixeer de variabele met nummer xx de waarde yyyy.
  - rel vaxx: geef de variabele met nummer xx weer vrijdag
  - get paxx: laat de waarde zien van parameter xx
  - set paxx yyyy: zet parameter met nummer xx op de waarde yyyy en sla dit op in eeprom.
  - get all\_vars: laat de waarden zien van alle variabelen
  - get all\_pars: laat de waarden zien van alle parameters
  - auto\_tune: activeer de auto tuning functie
  - set\_factory\_defaults: initialiseer alle parameters op hun standaard waarde
  - set\_mash <nr> <temp> <tijd>: zet temp\_tijd paar <nr> op <temp> en <tijd>
  - ver: laat het versienummer zien van de hobbybrouwen firmware
- Koudevraag: De koudevraag is een signaal waarmee een koeling geactiveerd kan worden. Het is een signaal van 220 Volt AC waarmee maximaal 1000 Watt geschakeld kan worden. Om te voorkomen dat een koeling te vaak aan- en uitgezet wordt, wordt deze op tijd geregeld. Als de waarde van de regelaar boven een bepaald minimum komt (typical 60 %), wordt de koeling aangezet. Komt de waarde van de regelaar onder een bepaald minimum (typical 40 %), dan wordt de koeling weer uitgezet.
- Warmtevraag: De warmtevraag is een signaal waarmee het verwarmingselement van energie wordt voorzien. Het is een signaal van 220 Volt AC waarmee maximaal 3000 Watt aan het verwarmingselement toegevoerd wordt. De periodetijd is 5 seconden. Als de waarde van de regelaar bijvoorbeeld 60 % warmtevraag aangeeft, dan wordt het verwarmingselement 3 seconden aangezet en 2 seconden uitgezet.
- Gasklep stand: De gasklep\_stand is een signaal tussen 0 en 100 %. Het is een pulsbreedte gemoduleerd (PWM) signaal. De frequentie van dit signaal is 20-25 kHz. Door de pulsbreedte te variëren, wordt de gasklep verder/minder ver open gezet, zodat de hoeveelheid gas geregeld wordt.
- Commands Display: De commando's zoals de gebruiker die vanaf het display van de PID regelaar kan invoeren m.b.v. druktoetsen.

- **Display Info:** De informatie die op het display getoond wordt aan de gebruiker. Er kan een maximum van 2 signalen tegelijkertijd getoond worden, zoals bijv. de referentie temperatuur en de actuele temperatuur.

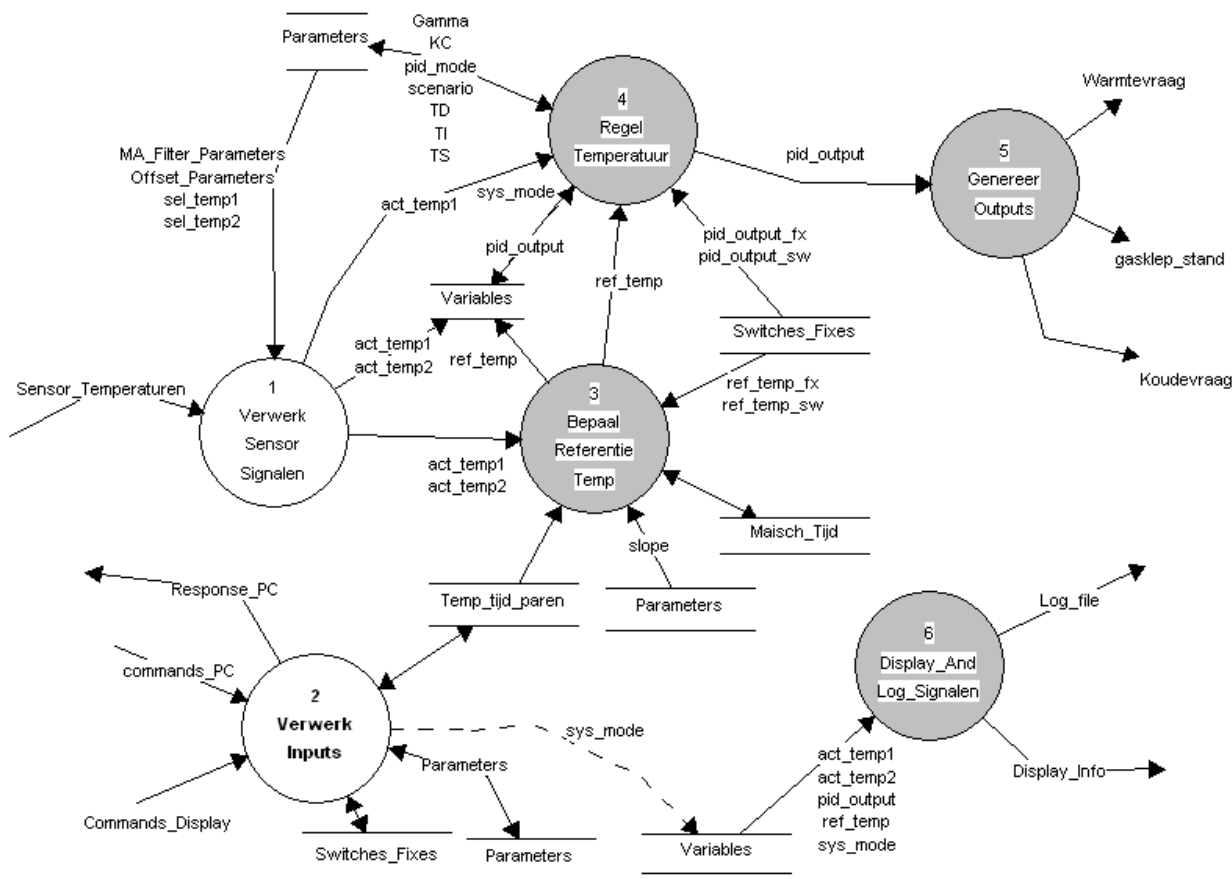
## 2.2 Ontwerpbesluiten

De volgende ontwerpbesluiten zijn genomen:

- Er zal gewerkt gaan worden met een ARM7 32 bits microcontroller, die al gemonteerd is op een print. Dit is de SAM7-H64 van Olimex (zie <http://www.olimex.com/dev/sam7-h64.html>). In 2<sup>e</sup> instantie kan gekeken worden in hoeverre het kosteneffectiever is om deze microcontroller te integreren in het bestaande ontwerp.
- Het te gebruiken ontwikkelsysteem is van Crossworks (zie <http://www.rowley.co.uk/arm/index.htm>).

## 2.3 Product / System Componenten

Figuur 2 laat het context diagram zien van de PID regelaar. Door nu in te zoomen op bol 0: PID Regelaar ontstaat een onderliggend data-flow diagram dat de belangrijkste deelfuncties van het totale systeem laat zien. Dit is weergegeven in figuur 2:



**Figuur 2: DFD 0: PID Regelaar**

De diverse deelcomponenten zullen in de komende paragraaf op hoofdlijnen beschreven worden. Voor de verdere ontwerpdetails wordt verwezen naar de betreffende paragraaf in hoofdstuk 3 "Detail Ontwerp".

## 2.4 **Beschrijving van de software deelcomponenten**

In deze paragraaf wordt een korte beschrijving gegeven van iedere deelcomponent. De verdere uitwerking van iedere deelcomponent wordt in hoofdstuk 3 "Detail Ontwerp" gedaan.

### 2.4.1 **Deelcomponent 1: Verwerk Sensor signalen**

Deze functie verwerkt de informatie die afkomstig is van de temperatuursensoren. Dit kunnen 4 soorten sensoren zijn: een PT100 sensor, een thermokoppel sensor, 1 tot 2 I2C temperatuursensoren en 1 tot 2 One-Wire temperatuursensoren.

Aan de hand van de parameters `sel_temp1` en `sel_temp2` wordt geselecteerd welke sensorwaarde aan `act_temp1` of `act_temp2` wordt gekoppeld. Hiermee wordt het mogelijk om iedere sensorwaarde op `act_temp1` of `act_temp2` te plaatsen.

Ook het filteren van `act_temp1` en `act_temp2` gebeurt in deze functie. De filtering wordt verzorgd middels moving-average filters, waarvan de orde in te stellen is (middels `MA_Filter_Parameters`). Hoe groter die parameter, des te zwaarder wordt er gefilterd. Daarnaast is er de mogelijkheid om een offset bij de temperatuur op te tellen (via `Offset_Parameters`).

### 2.4.2 **Deelcomponent 2: Verwerk Inputs**

Deze functie verwerkt zowel de inputs van de gebruiker via het display van de PID regelaar, alsmede de inputs die de gebruiker via de PC kan invoeren.

Als een PC is aangesloten, dan heeft deze prioriteit boven het display van de PID regelaar. Invoer die gedaan wordt via het display van de PID regelaar, wordt in dat geval dan ook genegeerd.

Deze functie heeft de volgende mogelijkheden:

- Parameters op vragen en deze te wijzigen.
- Variabelen fixeren door deze een vaste waarde te geven (`Switches_Fixes`). Zie ook paragraaf 2.6
- Temperatuur-tijd paren van het maischtraject kunnen instellen en wijzigen.

### 2.4.3 **Deelcomponent 3: Bepaal Referentie Temp**

Deze functie bepaalt de referentietemperatuur voor de PID-regelaar. Deze referentietemperatuur wordt bepaald uit de gedefinieerde temperatuur-tijd paren.

### 2.4.4 **Deelcomponent 4: Regel Temperatuur**

Dit is de feitelijke PID-regelaar, die het regelen voor zijn rekening neemt.

Maar ook de auto-tuning functie (het bepalen van de optimale regelparameters) is een belangrijke functie van deze component.

### 2.4.5 **Deelcomponent 5: Genereer Outputs**

Deze functie genereert de signalen voor zowel het verwarmingselement, de koeling als voor de modulerende gasklep.

### 2.4.6 **Deelcomponent 6: Display en Log Signalen**

Deze functie toont een aantal signalen op het display van de PID regelaar EN het genereert en verstuurt log-informatie naar de PC (iedere 5 seconden).

## 2.5 **Beschrijving van de hardware deelcomponenten**

In deze paragraaf wordt een korte beschrijving gegeven van iedere hardware deelcomponent. De indeling van de hardware componenten is redelijk eenvoudig en zal uit de volgende componenten bestaan:

- Een voedingsprint: deze zal de noodzakelijke spanningen leveren. Op dit moment wordt voorzien in +24 Vdc (10 VA) voor de gasklep, +5 V (1 A) voor de microcontroller en + en – 3.3 Volt.
- Een triac print: deze schakelt het verwarmingselement en een 2<sup>e</sup> (identieke) triac print schakelt de koeling. Dit wordt een aparte print, omdat deze grote vermogens gaat schakelen
- Een display print: deze zal de 7-segment displays gaan bevatten en de knoppen en LEDs
- Een hoofdprint: hier wordt de Olimex microcontroller print op gemonteerd. Ook de resterende hardware kan hier een plaats op krijgen.

Voor meer details wordt verwezen naar hoofdstuk 3 “Detail Ontwerp”.

## 2.6 Beschrijving van de Interfaces

Voor een beschrijving van de interfaces wordt verwezen naar de data-dictionary in Appendix I. Hier wordt van iedere data-flow (ieder signaal) een omschrijving gegeven.

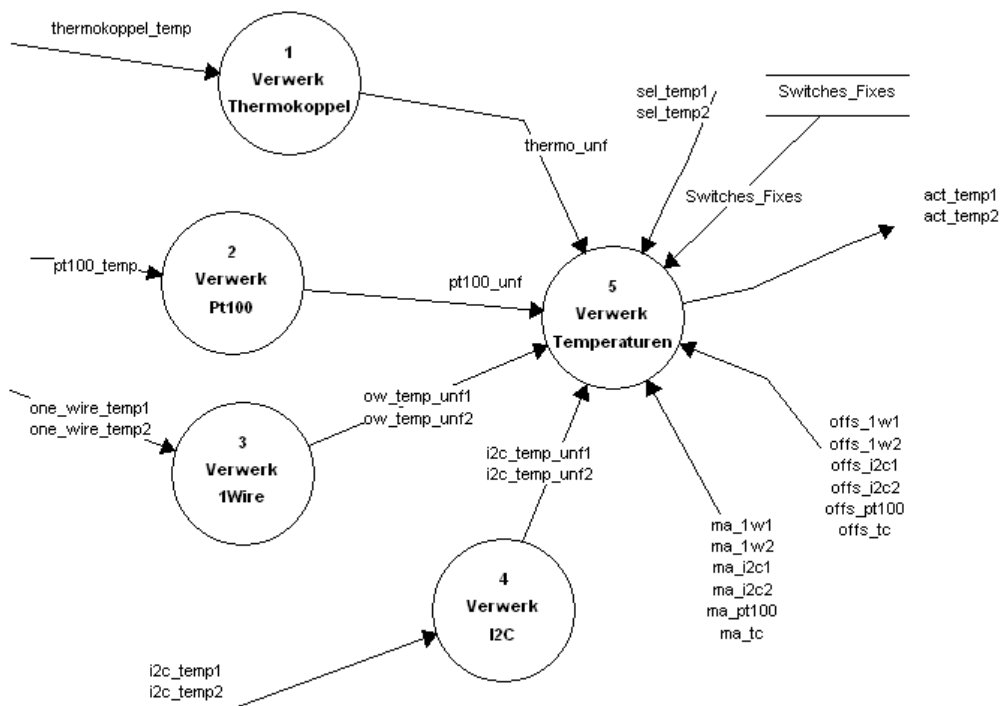
Enige nadere uitleg behoeven de data-stores uit figuur 2, omdat deze cruciaal zijn voor het begrip van het gehele systeem:

- Switches Fixes: om het systeem goed te kunnen testen, maar ook voor flexibiliteit tijdens het gebruik, bestaat er de mogelijkheid om een signaal een vaste waarde te geven. Dit gebeurt door van zo'n signaal een switch waarde (met `_sw` als toevoeging in de naam) te activeren. Tegelijkertijd kan de te fixeren waarde (met `_fx` als toevoeging in de naam) een waarde krijgen. In pseudo-code:  
*Als <variabele>\_sw is true  
Dan <variabele> = <variabele>\_fx  
Anders <variabele> wordt op de normale manier bepaald*
- Parameters: de parameters bevatten alle instellingen van het systeem en worden opgeslagen in niet-vluchtig geheugen (eeprom).
- Variables: de variabelen bevatten de belangrijkste signalen van het systeem
- Temp Tijd Paren: dit is een tabel met informatie over het aantal minuten dat het systeem een bepaalde temperatuur dient vast te houden.
- Maisch Tijd: dit is de verstreken tijd sinds de start van een bereikte temperatuur

### 3 Detail Ontwerp

#### 3.1 Deelcomponent 1: Verwerk Sensor signalen

Deze deelcomponent is weer verdeeld in een aantal deelcomponenten, conform figuur 3:



**Figuur 3: DFD 1: Verwerk Sensor Signalen**

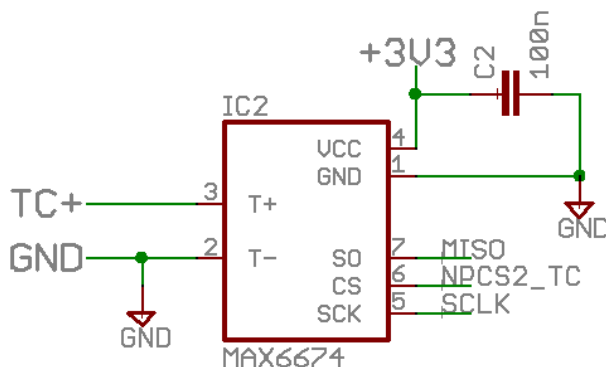
Deze deelcomponenten worden in de hierna volgende paragrafen verder uitgewerkt:

##### 3.1.1 Deelcomponent 1.1: Verwerk Thermokoppel

We gaan uit van een K-type thermokoppel (zie requirements R8d). Deze heeft een gevoeligheid van 40.6  $\mu\text{V}/^\circ\text{C}$ .

De interface wordt geheel verzorgd door een speciaal IC hiervoor, de MAX6674. Deze communiceert met de microcontroller via de SPI bus. Het temperatuurbereik is in dit geval 0 .. +128  $^\circ\text{C}$ . Indien een groter bereik noodzakelijk zou zijn, dan kan nog gekozen worden voor de MAX6675, deze heeft een temperatuurbereik tot 1024  $^\circ\text{C}$ . Maar voor de hobbybrouwen PID regelaar voldoet de MAX6674 prima.

Dit leidt tot het volgende schema:



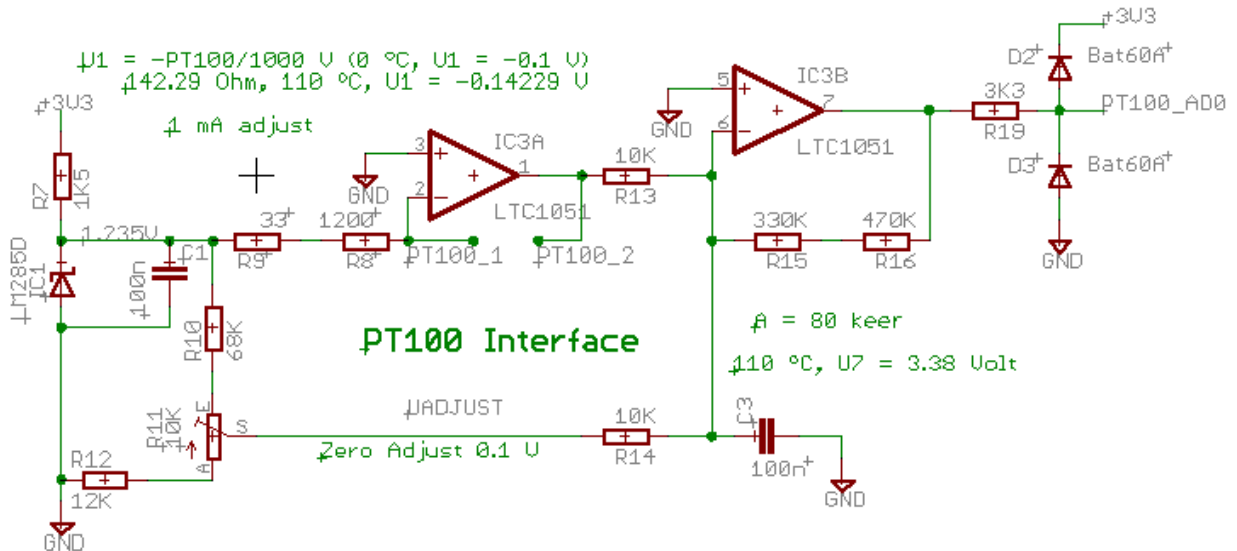
**Figuur 4: Schema van themokoppel circuit**

Op de ingangen T+ en T- wordt het K-type thermokoppel aangesloten. Het IC is ook in staat te detecteren of er wel of geen thermokoppel is aangesloten.

### 3.1.2 Deelcomponent 1.2: Verwerk PT100

Een PT100 element kent een weerstand van precies 100 Ω bij 0 °C. Deze weerstand wordt groter naarmate de temperatuur stijgt. Om de PT100 temperatuursensor te kunnen gebruiken (zie requirements R8c), moet er een kleine stroom (ongeveer 1 mA) door de sensor gestuurd worden. Onderstaand schema is gebaseerd op een ontwerp van Tito Smailigich, zie <http://www.edn.com/contents/images/41201di.pdf>, laatste pagina.

Dit leidt tot het volgende schema:



Figuur 5: Schema van PT100 circuit

De LM285D spanningsreferentie levert een nauwkeurige 1.235 V af. M.b.v. de weerstanden R8 en R9 is de stroom vrijwel gelijk aan 1 mA. Dit is ook de stroom door de PT100 sensor. Bij een temperatuur van 0 °C is de weerstand van een PT100 element precies 100 Ω. De spanning op de uitgang van IC3A is dan gelijk aan  $(-100/1233) \cdot 1.235 \text{ V} = -0.1 \text{ Volt}$ . Door een nauwkeurige 100 Ω weerstand aan te sluiten (i.p.v. de PT100 sensor) kan de uitgangsspanning (pin 7 van IC3B) op 0 Volt afgeregeld worden, m.b.v. potmeter R11.

Door vervolgens een weerstand van 142 Ω (komt overeen met een temperatuur van 110 °C) aan te sluiten, wordt de uitgang van IC3A gelijk aan  $(-142/1233) \cdot 1.235 = -0.142 \text{ V}$ . Op pin 6 van IC3B blijft dan -42 mV over. IC3B is ingesteld als een Inverterende versterker met een versterking van  $-800\text{k}/10\text{k} = -80 \times$ . De uitgang wordt dan gelijk aan  $-80 \times -42 \text{ mV} = 3.36 \text{ Volt}$ .

Om de ingang van de AD-converter te beschermen tegen te hoge en te lage spanningen, zijn D2 en D3 aangebracht. Deze dioden hebben een  $U_f$  van 0.12 V, waardoor de ingang van de AD-converter niet lager kan worden dan -0.12 V (-0.3 V is de absolute maximum rating). Indien we willen zorgen dat bij deze minimale spanning er maximaal een stroom van 1 mA door de opamp geleverd gaat worden, dan moet de weerstand R19 minimaal  $(3.3-0.12) / 1 \text{ mA} = 3.18 \text{ k}\Omega$  bedragen. Gekozen wordt hier voor een weerstand van 3.3 kΩ. Dit wordt dan ook de uitgangsimpedantie van het circuit dat de AD-converter aanstuurt.

In de datasheet van de SAM7 wordt een maximale uitgangsimpedantie gegeven (zie pag. 508). Hier staat dat  $Z_{out} \leq (\text{SHTIM}(\text{ns})-589) \times 7.69$  moet zijn. Invullen levert een SHTIM(ns) op van minimaal 1018 nsec. Middels de SHTIM parameter in het register kan deze minimum tijd ingesteld worden. De formule hiervoor (zie pag. 489) is:  $1018 \text{ nsec.} \leq (\text{SHTIM}+1)/\text{ADCClock}$ . De ADCClock wordt ingesteld op  $\text{MCK} / ((\text{PRESCAL}+1) \cdot 2) =$

$47923200 / 10 = 4.79$  MHz Hz. Hiermee dient SHTIM minimaal ingesteld te worden op 4. Gekozen wordt voor de waarde 5, om aan de veilige kant te blijven.

### 3.1.3 Deelcomponent 1.3: Verwerk 1Wire

Deze deelcomponent leest de actuele waarden uit van de temperatuursensoren, die aangesloten zijn op de One-Wire bus. Er kunnen maximaal 2 temperatuursensoren (bijv. de DS1820) tegelijk aangesloten zijn (zie requirement R8a).

Voor het inlezen van waarden van een DS1820 temperatuursensor wordt gebruik gemaakt van een (software) library van Martin Thomas (zie <http://www.siawawi.arubi.uni-kl.de/avr-projects>). Deze library bestaat uit de volgende files:

- ds18x20.c en ds18x20.h
- onewire.c en onewire.h

De one-wire lijn wordt aangesloten op poort PA1 van de microcontroller. Deze dient via een weerstand van 4K7 verbonden te zijn met de +3.3 V voedingsspanning.

### 3.1.4 Deelcomponent 1.4: Verwerk I2C

Deze deelcomponent leest de actuele waarden uit van de temperatuursensoren, die aangesloten zijn op de I2C bus (maximaal 2, zie requirement R8b). Omdat de I2C bus werkt met vaste adressen, zijn de volgende adressen gedefinieerd en in te stellen met de parameters `i2c_addr1` (PA65) en `i2c_addr2` (PA66):

<code>i2c_addr1</code> , <code>i2c_addr2</code>	I2C adres	Omschrijving
<b>0</b>	0x90 / 0x91	LM75 of LM92 externe temperatuursensor
<b>1</b>	0x92 / 0x93	LM75 of LM92 externe temperatuursensor
<b>2</b>	0x94 / 0x95	LM75 of LM92 externe temperatuursensor
<b>3</b>	0x96 / 0x97	LM75 of LM92 externe temperatuursensor

De AT91SAM7S64 microcontroller kent een Two-Wire interface (TWI), die gebruikt kan worden om de I2C componenten aan te sturen. Aandachtspunten hierbij zijn:

- De TWI-clock moet ingesteld worden, uitgaande van de master-clock van de  $\mu$ C. Dit gebeurt via het `TWI_CWGR` register.
- De  $\mu$ C wordt ingesteld als de master, de aangesloten ICs als de slave
- Er wordt een routine `i2c_write()` en een routine `i2c_read()` gemaakt. Beide routines kunnen meerdere bytes verwerken, zie ook de flow-charts in de user guide van de  $\mu$ C.

### 3.1.5 Deelcomponent 1.5: Verwerk Temperaturen

Deze deelcomponent voert de volgende stappen uit (ook in deze volgorde):

1) Filteren van alle temperaturen. Voor de (laagdoorlaat)filtering wordt een moving-average filter genomen. Van iedere temperatuur worden de volgende parameters gebruikt:

PA20	ma_pt100	Orde voor pt100 temperatuur
PA21	ma_tc	Orde voor thermokoppel temperatuur
PA22	ma_i2c1	Orde voor 1 <sup>e</sup> I2C temperatuur
PA23	ma_i2c2	Orde voor 2 <sup>e</sup> I2C temperatuur
PA24	ma_1w1	Orde voor 1 <sup>e</sup> one-wire temperatuur
PA25	ma_1w2	Orde voor 2 <sup>e</sup> one-wire temperatuur

2) Toevoegen van een offset aan iedere temperatuur. Voor de offset worden de volgende parameters gebruikt:

PA26	offs_pt100	Offset in E-1 °C voor pt100 temperatuur
PA27	offs_tc	Offset in E-1 °C thermokoppel temperatuur
PA28	offs_i2c1	Offset in E-1 °C voor 1 <sup>e</sup> I2C temperatuur
PA29	offs_i2c2	Offset in E-1 °C voor 2 <sup>e</sup> I2C temperatuur
PA30	offs_1w1	Offset in E-1 °C voor 1 <sup>e</sup> one-wire temperatuur
PA31	offs_1w2	Offset in E-1 °C voor 2 <sup>e</sup> one-wire temperatuur

3) Fixeren van de temperaturen. Met behulp van de switches (\_sw extensie) en fixes (\_fx extensie) kunnen alle zes de gemeten temperaturen een waarde opgedrukt krijgen.

4) Bepalen van de twee temperaturen (act\_temp1 en act\_temp2) voor verder gebruik. Met behulp van de parameters sel\_temp1 (PA32) en sel\_temp2 (PA33) wordt de gewenste variabele voor act\_temp1 en act\_temp2 bepaald. Hierbij wordt de volgende codering gedefinieerd:

0	pt100 temperatuur
1	thermokoppel temperatuur
2	1 <sup>e</sup> I2C temperatuur
3	2 <sup>e</sup> I2C temperatuur
4	1 <sup>e</sup> one-wire temperatuur
5	2 <sup>e</sup> one-wire temperatuur

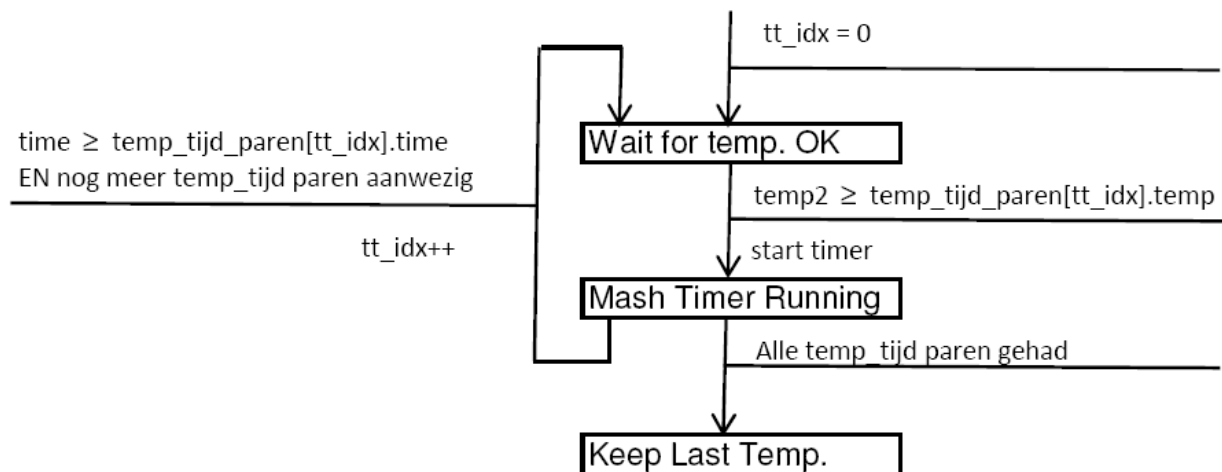
### 3.2 Deelcomponent 2: Verwerk Inputs

Nog in te vullen

### 3.3 Deelcomponent 3: Bepaal Referentie Temp

Bij opstarten van het systeem wordt een index in de temperatuur-tijden tabel (tt\_idx) geïnitieerd op 0. Deze functie genereert een temp\_ref (temperatuur referentie voor de PID regelaar) door iedere keer de temperatuurwaarde uit de temperatuur-tijden tabel te lezen, die bij de index tt\_idx hoort.

Daarna wordt gekeken of de werkelijke temperatuur (act\_temp2) deze referentietemperatuur reeds bereikt heeft. Is dit het geval, dan wordt er een timer gestart. Wanneer deze timer groter is dan de tijd uit de temperatuur-tijden tabel, dan wordt het volgende temperatuur-tijd paar uit de tabel gehaald. Het toestandsdiagram



**Figuur 6: Toestandsdiagram voor bepaling referentie temperatuur**

Als laatste wordt de referentie temperatuur (temp\_ref) nog voorzien van een switch en fix waarde, zodat deze referentie ook handmatig in te stellen is.

### 3.4 Deelcomponent 4: Regel Temperatuur

### 3.5 Deelcomponent 5: Genereer Outputs

### 3.6 Deelcomponent 6: Display en Log Signalen

4 Requirements Traceability

	DFD1 §2.3.1	DFD2 §2.3.2	DFD3 §2.3.3	DFD4 §2.3.4	DFD5 §2.3.5	DFD6 §2.3.6
R1			X			
R2		X	X			
R3			X			
R4		X		X		
R5		X				
R6		X				
R7	X					
R8	X					
R9				X		
R10					X	
R11					X	
R12						
R13		X				
R14						X
R15						X
R16						X
R17					X	
R18						

○Oud: nog aanpassen aan nieuwe versie van specificatie!

**Appendix 1. AT91SAM7S64 PIO Port Mapping**

PIO Controller A				Application Usage
I/O Line	Periph. A	Periph. B	Comments	Function
PA0	<b>PWM0</b>	TIOA0	High-Drive	Gas Valve Output 28V/25 kHz PWM
<b>PA1</b>	PWM1	TIOB0	High-Drive	One-Wire DQ line
PA2	<b>PWM2</b>	SCK0	High-Drive	Triac1 Output
PA3	<b>TWD</b>	NPCS3	High-Drive	I2C SDA
PA4	<b>TWCK</b>	TCLK0		I2C SCL
PA5	<b>RXD0</b>	NPCS3		RS232 Port 0 Receive
PA6	<b>TXD0</b>	PCK0		RS232 Port 0 Transmit
<b>PA7</b>	RTS0	PWM3		Triac2 Output (as of V2.4)
<b>PA8</b>	CTS0	ADTRG		Alive LED
PA9	DRXD	NPCS1		
PA10	DTXD	NPCS2		
PA11	<b>NPCS0</b>	PWM0		MAX6951 Chip Select (SPI) (Display)
PA12	<b>MISO</b>	PWM1		MCP23S08_SO (IO)
PA13	<b>MOSI</b>	PWM2		MAX6951_DIN (Display) + MCP23S08_SI (IO)
PA14	<b>SPCK</b>	PWM3		SPI Clock
PA15	TF	TIOA1		
PA16	TK	TIOB1		
PA17	TD	PCK1	<b>AD0</b>	PT100 Analog Input
PA18	RD	PCK2	AD1	
PA19	RK	FIQ	AD2	
PA20	RF	IRQ0	AD3	
PA21	<b>RXD1</b>	PCK1		RS232 Port 1 Receive
PA22	<b>TXD1</b>	NPCS3		RS232 Port 1 Transmit
PA23	SCK1	PWM0		
PA24	RTS1	PWM1		
PA25	CTS1	PWM2		
PA26	DCD1	TIOA2		
PA27	DTR1	TIOB2		
<b>PA28</b>	DSR1	TCLK1		MISO_MAX6674, serial data
<b>PA29</b>	RI1	TCLK2		CLK_MAX6674, clock signal
<b>PA30</b>	IRQ1	NPCS2		CS_MAX6674 Chip Select, Thermocouple
PA31	<b>NPCS1</b>	PCK2		MCP23S08 Chip Select (SPI)

**Appendix 2. Parameters and Variables List**

<b>Menu 1 [MNU1]: PID Control menu (eeprom page 0)</b>					
Number	Name	Unity	Values	Default	Description
PA01	Kc	% / °C	[0, 999]	80	Kc parameter for PID controller
PA02	Ti	sec.	[0, 999]	490	Ti parameter for PID controller
PA03	Td	sec.	[0, 999]	45	Td parameter for PID controller
PA04	Gamma	sec.	[0, 999]	4	Filter time-constant for D-action
PA05	Ts	E-1 sec.	[1,50]	10	Sample-time (time between 2 PID calls)
PA06	slope1	E-2 °C / sec.	[0, 9999]	100	Slope for reference temperature
PA07	pid_mode	-	[0,1]	0	0=PID takahashi; 1=self-tuning PID
PA08	scenario	-	[0,1,2]	0	0=standard, 1=HERMS, 2=climate-control
PA09	slope2	E-2 °C / sec.	[0, 9999]	100	Slope for temperatures from sensors
<b>Menu 2 [MNU2]: Input menu (eeprom page 1)</b>					
Number	Name	Unity	Values	Default	Description
PA20	ma_pt100	-	[0,20]	10	Order of MA filter for PT100 temperature
PA21	ma_tc	-	[0,20]	10	Order of MA filter for thermocouple temp.
PA22	ma_i2c1	-	[0,20]	5	Order of MA filter for I2C temp. sensor 1
PA23	ma_i2c2	-	[0,20]	5	Order of MA filter for I2C temp. sensor 2
PA24	ma_1w1	-	[0,20]	5	Order of MA filter for One-Wire temp. sensor 1
PA25	ma_1w2	-	[0,20]	5	Order of MA filter for One-Wire temp. sensor 2
PA26	offs_pt100	E-1 °C	[-128,+127]	0	Offset to add to PT100 temperature
PA27	offs_tc	E-1 °C	[-128,+127]	0	Offset to add to thermocouple temperature
PA28	offs_i2c1	E-1 °C	[-128,+127]	0	Offset to add to I2C temperature 1
PA29	offs_i2c2	E-1 °C	[-128,+127]	0	Offset to add to I2C temperature 2
PA30	offs_1w1	E-1 °C	[-128,+127]	0	Offset to add to One-Wire temperature 1
PA31	offs_1w2	E-1 °C	[-128,+127]	0	Offset to add to One-Wire temperature 2
PA32	sel_temp1	-	[0,8]	0	Variable to use for temp1, see var. List
PA33	sel_temp2	-	[0,8]	6	Variable to use for temp2, see var. List
PA34	nr_temp	-	[0,1]	0	0=use only temp1 for control and progress 1=temp1 for control, temp2 for progress
PA35	outputs	-	[0,7]	2	0=none, 1=gas valve, 2=Triac1, 4=Triac2, 7=all
<b>Menu 3 [MNU3]: Temperature-Time menu (eeprom page 2)</b>					
Number	Name	Unity	Values	Default	Description
PA40	time0	min.	[0,31680]	3	First time-temp. pair: 31680 = 22 days x 24 hours x 60 minutes
PA41	temp0	°C	[0,100]	20	
PA42	time1	min.	[0,31680]	0	Second time-temp. pair
PA43	temp1	°C	[0,100]	0	
PA44	time2	min.	[0,31680]	0	Third time-temp. pair
PA45	temp2	°C	[0,100]	0	
PA46	time3	min.	[0,31680]	0	Fourth time-temp. pair
PA47	temp3	°C	[0,100]	0	
PA48	time4	min.	[0,31680]	0	Fifth time-temp. pair
PA49	temp4	°C	[0,100]	0	
PA50	time5	min.	[0,31680]	0	Sixth time-temp. pair
PA51	temp5	°C	[0,100]	0	
PA52	time6	min.	[0,31680]	0	Seventh time-temp. pair
PA53	temp6	°C	[0,100]	0	
PA54	time7	min.	[0,31680]	0	Eight time-temp. pair
PA55	temp7	°C	[0,100]	0	
<b>Menu 4 [MNU4]: Hardware menu (eeprom page 3)</b>					
PA65	i2c_addr1	-	[0,3]	0	Which LM92 device for I2C temp. sensor 1
PA66	i2c_addr2	-	[0,3]	1	Which LM92 device for I2C temp. sensor 2
PA67	use_usb_lcd	-	[0,3]	0	0=none, 1=USB only, 2=LCD only, 3=both

PA68	var_led1	-	[0,8]	0	Variable to display on Top display, see var. List
PA69	var_led2	-	[0,8]	6	Variable to display on Bottom display, see var. List
PA70	led_intens	-	[0,15]	4	Intensity for LED display (0=off)
<b>Menu 5 [MNU5]: Variables menu (NOT stored in eeprom)</b>					
VA01	pt100_temp	°C	[0,100]	0	Temp. from PT100 temp. sensor
VA02	tc_temp	°C	[0,100]	0	Temp. from thermocouple temp. sensor
VA03	i2c1_temp	°C	[0,100]	0	Temp. from 1 <sup>st</sup> I2C temp. sensor
VA04	i2c2_temp	°C	[0,100]	0	Temp. from 2 <sup>nd</sup> I2C temp. sensor
VA05	one_wire_temp[0]	°C	[0,100]	0	Temp. from 1 <sup>st</sup> one-wire temp. sensor
VA06	one_wire_temp[1]	°C	[0,100]	0	Temp. from 2 <sup>nd</sup> one-wire temp. sensor
VA07	temp_ref	°C	[0,100]	0	Reference temp. (the setpoint-value SP)
VA08	pid_output	%	[0,100]	0	PID controller output

### Appendix 3. How to operate the Menu and the Display

The display works together with the built-in menu. There are two main functions:

- Displaying of variables on the display (these variables are typically the measured temperature and the reference temperature, but other variables can also be selected)
- Changing the value of any of the parameters within the menus (see appendix 2 for a complete list of the menus and the parameters)

The menu contains the following modes:

- **NORMAL mode**: after power-up has finished, the display always starts in normal mode. In this mode, the two selected variables are shown on the display.
  - If the **DOWN** and **RIGHT** buttons are pressed simultaneously, the factory default settings are restored.
  - If the **DOWN** and **LEFT** buttons are pressed simultaneously, the display test mode is enabled (all LEDs are on).
  - If the **UP** button is pressed, the MENU SELECT mode is activated.
- **MENU SELECT mode**: to access this menu, press the **UP** button when in NORMAL mode. One of the following menus is shown (which corresponds to the menus listed in Appendix 2):

PBL1
Pid

PBL2
InOu

PBL3
TeTi

PBL4
Hw

PBL5
SwF

- To change between any of these menus, press the **UP** or the **DOWN** button.
- If the desired menu is shown, it can be selected by pressing the **RIGHT** button. This will activate the PARAMETER SELECT mode.
- If no keys are pressed for a certain time (typically 10 seconds), the NORMAL MODE of operation is restored again.
- **PARAMETER SELECT mode**: to access this menu, press the **RIGHT** button in the MENU SELECT mode.
  - The TOP display shows the parameter number, which is identical to the numbers listed in appendix 2 (Parameters and Variables List). Example:

PA65
0144
  - To change between any of these parameters within this menu, press the **UP** or the **DOWN** button (they may be pressed continuously).
  - If the **LEFT** button is pressed, the MENU SELECT mode is activated again.
  - If the desired parameter is shown, it can be selected by pressing the **RIGHT** button. This will activate the PARAMETER CHANGE mode.
  - If no keys are pressed for a certain time (typically 10 seconds), the NORMAL MODE of

operation is restored again.

- PARAMETER CHANGE mode: to access this menu, press the **RIGHT** button in the PARAMETER SELECT mode. The lowest digit of the actual value starts to blink, indicating that this digit can be changed.
  - Press the **LEFT** or **RIGHT** button to change the digit to blink.
  - If the **LEFT** button is pressed one time more than the maximum number of digits, all digits start to blink. This is an indication that the PARAMETER SAVE mode is entered. The PARAMETER CHANGE mode can be entered again by pressing the **RIGHT** button. This is indicated by blinking of a single digit instead of all digits.
  - By pressing the **UP** or **DOWN** button, the blinking digit can be increased or decreased.
  - If no keys are pressed for a certain time (typically 10 seconds), the NORMAL MODE of operation is restored again.
- PARAMETER SAVE mode: to access this menu, press the **LEFT** button one time more than the maximum number of digits when in the PARAMETER CHANGE mode. In this mode all digits blink together (approx. two times per second).
  - The **LEFT** button has to be released and then pressed again to save all the parameters of this menu (but **NOT** all parameters of all menus!) into non-volatile memory (eeprom). The display will show the following text, before returning to the PARAMETER SELECT mode:

EEP
SAVE
  - If the **RIGHT** button is pressed, the PARAMETER SAVE mode is abandoned and the PARAMETER CHANGE mode is selected again.
  - If no keys are pressed for a certain time (typically 10 seconds), the NORMAL MODE of operation is restored again.

#### Appendix 4. RS232 Serial port 0: available commands

Connect a 9 pin serial connector to serial port 0 of the PID controller. Set communication parameters to 115200, 8, N,1.

The following commands are available:

<b>get</b>	get paxx	xx is the parameters number, see appendix 2 for details. Example: <i>get pa65</i> to retrieve the value of parameter 65 (i2c_addr1)
	get vaxx	xx is the variable number, see appendix 2, menu 5 for details. Example: <i>get va01</i> to retrieve the value of variable 01 (pt100_temp)
	get all_pars	this command lists the values of all available parameters
	get all_vars	this command lists the actual values of all available variables, together with the status of each switch and fix.
<b>log</b>	log	Shows the status of logging to RS232 serial port 0 (on or off).
	log on	Enables logging. Every 5 seconds, the actual value of all variables is written to serial port 0.
	log off	Disables logging
<b>pid</b>	pid	Show the status of the PID controller (on or off).
	pid on	Enables the PID controller. The PID led on the front-panel will light
	pid off	Disables the PID controller. The PID led on the front-panel will dim
	pid restart	Restarts the temp. time trajectory from the beginning. Note that the actual status of the PID controller (on, off) is NOT changed!
<b>rel</b>	rel vaxx	xx is the variable number, see appendix 2, menu 5 for details. Releases the variable. The system will use the actual value again instead of the value set by the user.
<b>set</b>	set paxx yyyy	xx is the parameter number, see appendix 2 for details. Example: <i>set pa65 2</i> to set the value of parameter 65 (i2c_addr1) to 2.
	set vaxx yyyy	xx is the variable number, see appendix 2, menu 5 for details. Forces the variable to a value set by the user. Example: <i>set va01 25</i> to set the value of variable va01 (pt100_temp) to a fixed value of 25 degrees. The actual value is overruled by this command.
<b>ver</b>	ver	Shows the revision number of the firmware of the system