Inhaltsverzeichnis

1 Einleitung

Nachhaltigkeit spielt in der heutigen Zeit eine wichtige Rolle. Neben einer Reduktion des Stromverbrauchs in privaten Haushalten, interessieren sich auch Firmen für eine Möglichkeit regenerative Energien zu nutzen. In dieser Arbeit soll für einen Landwirtschaftsbetrieb, der sich auf die Milchproduktion spezialisiert hat, ein Simulator erstellt werden. Dieser Simulator soll feststellen, ob die Verwendung eines Eiswasserspeichers in Kombination von einer Photovoltaik (PV) Anlage sinnvoll ist. Dies soll den eigentlichen Energieverbrauch der Milchkühlung reduzieren, indem der Eiswasserspeicher vorgeschalten wird. Er soll die Milch für den eigentlichen Kühlvorgang vorkühlen und dadurch den Energieaufwand reduzieren.

2 Szenario

Im Landwirtschaftsbetrieb in Fuchshain wird ein Kühlsystem verwendet, um die Milch auf die richtige Temperatur zu kühlen. Dabei gelangt die gewonne Milch in einen Milchtank, welche dann heruntergekühlt wird. In den eigentlichen Kühlvorgang kann nicht eingegriffen werden, weswegen die Vorkühlung der Milch vor dem Milchtank durch den Eiswasserspeicher gekühlt werden muss. Dieser Eiswasserspeicher soll durch eine vorhandene PV Anlage gespeist werden.
Während der Produktion muss die Milch von 35 auf 4 Grad Celsius abgekühlt werden. Diese Kühlmaßname ist energieaufwändig (304342,5 kJ bei 2500 Liter Milch). Damit die Hauptkühlung entlastet werden kann, soll die Effizienz einer Vorkühlung durch einen Eiswasserspeicher ermittelt werden. Dieser Speicher würde eine Abflachung der Lastspitzen ermöglichen und somit dem Betrieb Geld einsparen. Da die Anschaffung eines Eiswasserspeichers kostspielig ist, soll vorerst durch eine SImulation die Rentabilität ermittelt werden. Dabei wird angenommen, dass eine Vorkühlung der Milch von 35 auf 17 Grad Celsius stattfindet.

2.1 Eiswasserspeicher

Der Eiswasserspeicher kann 164 kg Eis speichern. Er verfügt weiterhin über einen Kompressor der Marke Maneurop MT-22 mit 3,51 kW Leistung. Der Kompressor ist für die Erzeugung des Eises verantwortlich. Weiterhin befindet sich im Speicher die horzontale Kreiselpumpe CEA 70/3/A-V der Firma LOWARA. Die Ladezeit für eine komplette Beladung mit Eis wird mit sechs Stunden angegeben.
Abbildung 2.1 zeigt den Verbrauch der Anlage am 1. Februar 2015 ohne den Betrieb eines Eiswasserspeichers. Die zwei hohen Ausschläge gibt die Leistung der zwei Kühlanlagen wieder. Diese Last soll durch einen Eiswasserspeicher gemindert werden, indem eine Vorkühlung stattfindet. In Kombination mit der PV Anlage erzeugt der Eiswasserspeicher Kühlleistung, welche die eigentliche Last der Hauptkühlung mindern soll.

Abbildung 2.1: Verbrauch 1. Februar 2015

2.2 Lösung

Für die Realisierung des Simulators wurde ein Raspberry PI zur Verfügung gestellt. Dieser soll softwareseitig alle 15 Minuten einen Simulationsschritt durchführen. In einem Schritt wird festgestellt, ob der Speicher beladen oder entladen wird. Während den Schritten wird die benötigte Leistung anhand einer S0-Schnittstelle übertragen. Die Leistung für den Ent- und Beladevorgang wurden vorgegeben.
Die S0-Schnittstelle ist nach DIN EN 62053-31 spezifiziert und dient zur Übertragung von Verbrauchsmesswerten. Häufiger Einsatz ist die Gebäudeautomatisierung und findet in einer vielzahl von Messgeräten anwendung. Elektrische Impulse dienen als Maßeinheit für den Verbrauch eines Geräts. Diese Impulse müssen ein gewissen Muster aufweisen, damit sie als gültiger Impuls gewertet werden. Ein Impuls besteht aus einem High und Low Signal. High muss ebenso wie Low mindestens 30 Millisekunden lang sein und die auf bzw. absteigende Flanke muss kleiner als fünf Millisekunden sein. Weiterhin entsprechen 1000 Impulse die Leistung von 1000 Watt.

3 Raspberry PI

Mit dem Raspberry PI bietet die Raspberry PI Foundation einen Einplatinencomputer, der für Experimente und Entwicklung von Programmen geeignet ist. Sein günstiger Preis von weniger als 40 Dollar und die Größe einer Kreditkarte tragen zu seinem Erfolg bei. Weiterhin existieren verschiedene Varianten des Computers, welche sich hauptsächlich in der Rechenleistung unterscheiden. Ursprünglich sollte der Raspberry PI zum Experimentieren von Studenten verwendet werden, fand jedoch schnell Anklag in divseren anderen Bereichen außerhalb von Universitäten, u.a. der Automatisierung von Abläufen.
Für diese Arbeit wurde ein Raspberry PI 2 vorgeschrieben, da er eine kostengünstige Alternative darstellt, um hardwarenahe Software zu verwirklichen. Hierbei spielen besonders die General Purpose Input/Output Pins eine wichtige Rolle. Weiterhin verfügt der Raspberry PI über eine Ethernet Schnittstelle, welche zur Kommunikation dient. Da es sich um einen Einplatinencomputer und nicht um einen klassischen Mikrocontroller handelt, findet sich auf dem Raspberry PI ein Betriebssystems wieder, welches eine Erleichterung für die Softwareentwicklung darstellt. Hierzu zählt insbesondere die Umsetzung des TCPI/IP Stacks.

3.1 Betriebssystem

Die Betriebssysteme für den Raspberry PI sind zahlreich. Am häufigsten ist jedoch eine Linux-Distribution anzutreffen. In diesem Projekt kommt Raspbian zum Einsatz, ein auf Debian angepasstes Betriebssystem. Die Installation von Raspbian findet mit dem NOOBS-Installer, ein von der Raspberry PI Foundation bereitgestellter Installationsassistent, statt. Das Betriebssystem vereinfacht die Softwareentwicklung dahingehend, dass es verschiedene Schnittstellen für den Zugriff auf die Hardware anbietet. Die für dieses Projekt am wichtigsten Schnittstellen sind einerseits die GPIO Pins und andererseits der Zugriff auf das Netzwerk.

3.2 GPIO

Von den insgesamt 40 Pins sind 26 GPIO Pins. Bei den anderen handelt es sich Pins für die Stromversorgung. Abbildung 3.1 zeigt die Belegung der Pins des Raspberry PI 2. Durch die GPIO Pins wird in diesem Projekt die benötigte S0-Schnittstelle implementiert. Diese Schnittstelle sendet Impulse, welche einen Leistungswert repräsentieren.

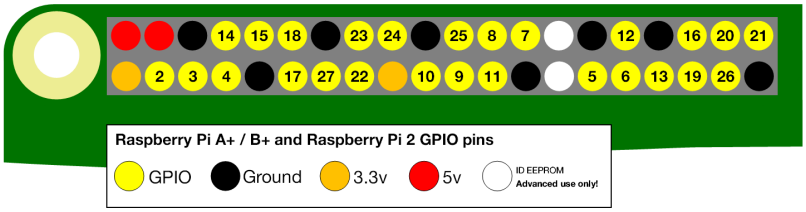

Abbildung 3.1: Pin Belegung Raspberry PI 2

Damit die GPIO Pins in einem C/C++ Programm angesprochen werden können, wird die Bibliothek *WiringPi* verwendet. Die Nummerierung der GPIO Pins unterschiedet sich zu der ursprünglichen Anordnung und ist aus Abbildung 3.2 ersichtlich.

P1: The Main GPIO connector							
WiringPi Pin	BCM GPIO	Name	Header		Name	BCM GPIO	WiringPi Pin
		3.3v	1	2	5v		
8	Rv1:0 - Rv2:2	SDA	3	4	5v		
9	Rv1:1 - Rv2:3	SCL	5	6	0v		
7	4	GPIO7	7	8	TxD	14	15
		0v	9	10	RxD	15	16
0	17	GPIO0	11	12	GPIO1	18	1
2	Rv1:21 - Rv2:27	GPIO2	13	14	0v		
3	22	GPIO3	15	16	GPIO4	23	4
		3.3v	17	18	GPIO5	24	5
12	10	MOSI	19	20	0v		
13	9	MISO	21	22	GPIO6	25	6
14	11	SCLK	23	24	CE0	8	10
		0v	25	26	CE1	7	11
WiringPi Pin	BCM GPIO	Name	Header		Name	BCM GPIO	WiringPi Pin

Abbildung 3.2: WiringPi Pinanordnung

Listing 3.1 zeigt ein einfaches Programm, welches *WiringPi* verwendet. Zeile vier zeigt eine Initialiserungsprozedur, welche einmalig aufgerufen werden muss, um auf die GPIO Pins zuzugreifen. Anschließend wird in Zeile fünf angegeben, ob auf dem Pin null lesend oder schreibend zugegriffen wird. Anschließend wird in einer Endlosschleife abwechselnd der Pin auf *HIGH* oder *LOW* gesetzt, jeweils mit einer Pause von 500 Millisekunden.

```
1  #include <wiringPi.h>
```

```
2  int main (void)
3  {
4    wiringPiSetup () ;
5    pinMode (0, OUTPUT) ;
6    for (;;)
7    {
8      digitalWrite (0, HIGH) ; delay (500) ;
9      digitalWrite (0,  LOW) ; delay (500) ;
10   }
11   return 0 ;
12 }
```

Listing 3.1: WiringPi Beispiel

3.3 Boost

Die C++ Bibliothek *boost* bietet diverse Funktionalitäten für die Entwicklung an. In diesem Projekt wird *boost* dazu verwendet, um auf das Netzwerk zuzugreifen und um Nebenläufigkeit zu ermöglichen. Die Netzwerkfunktionalitäten werden für eine Client-Server Architektur verwendet, um den Simulator zu steuern. Der Client sendet Kommandos, um die Simulation zu beeinflussen. Dadurch lässt sich einstellen, ob kühlt oder geladen wird.
Threads für die Nebenläufigkeit finden bei den einzelnen Komponenten des Programm anklang. Da die Komponenten quasi parallel laufen müssen werden verschiedene Threads benötigt. Das Verarbeiten der Kommandos des Clients, die Simulation selbst und die Übertragung der Leistung anhand der S0-Schnittstelle laufen je in einem eigenen Thread.

4 Aufbau

Abbildung 4.1 zeigt den physischen Aufbau der Projekts. Auf der einen Seite ist der Raspberry PI erkennbar. Auf der anderen Seite befindet sich ein Arduino UNO, welcher einen Datenlogger darstellen soll. Da zur Zeit der Entwicklung der Datenlogger nicht vorhanden war, wurde mittels eines Arduino UNO dessen Funktionalitäten nachgebaut. Der Arduino UNO zählt die ankommenden Impulse und summiert diese auf. Per Konsole werden die Werte an den Entwickler gegeben, sodass er testen kann, ob die gewünschte Anzahl von Impulsen angekommen ist. Im Betrieb soll jedoch ein richtiger Datenlogger zum Einsatz kommen.
Der Kern des Aufbaus bildet der Optokoppler vom Typ KB Knighbright KB 817. Der Optokoppler trennt die beiden Stromkreise voneinander galvanisch. Diese Trennung sichert den Raspberry PI gegen zu hohen Spannungen ab und verhindert somit seine Zerstörung.

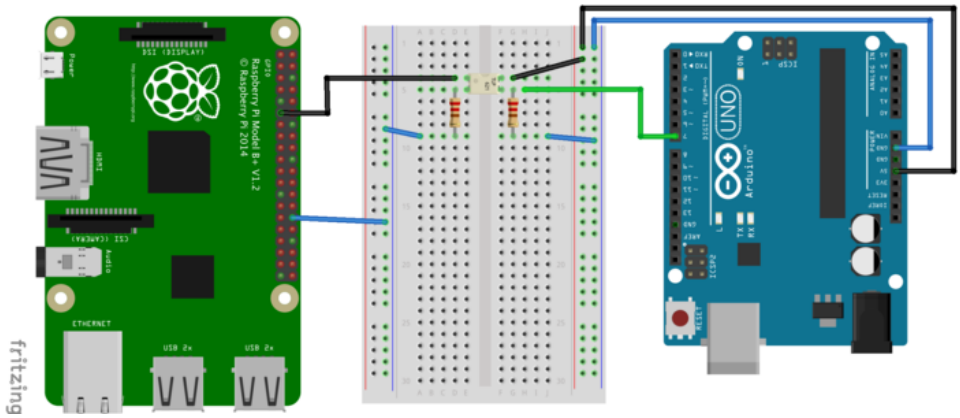

Abbildung 4.1: Schematischer Aufbai

Im eigentlichen Betrieb läuft der Simulator mit dem e.manager der Firma enerserve. Der e.manager erlaubt es Anlagen zu überwachen und den Verbrauch zu messen. Dies geschiet über diverse Schnittstellen, darunter auch die S0-Schnittstelle. Da der e.manager mit einer Spannung von 12 Volt betrieben wird, muss dieser galvanisch vom Raspberry

PI getrennt werden, da dieser mit 5 Volt betrieben wird. Dies geschiet mit dem bereits erwähnten Optokoppler. Abbildung 4.2 zeigt den Aufbau mit dem Raspberry PI und dem e.manager.

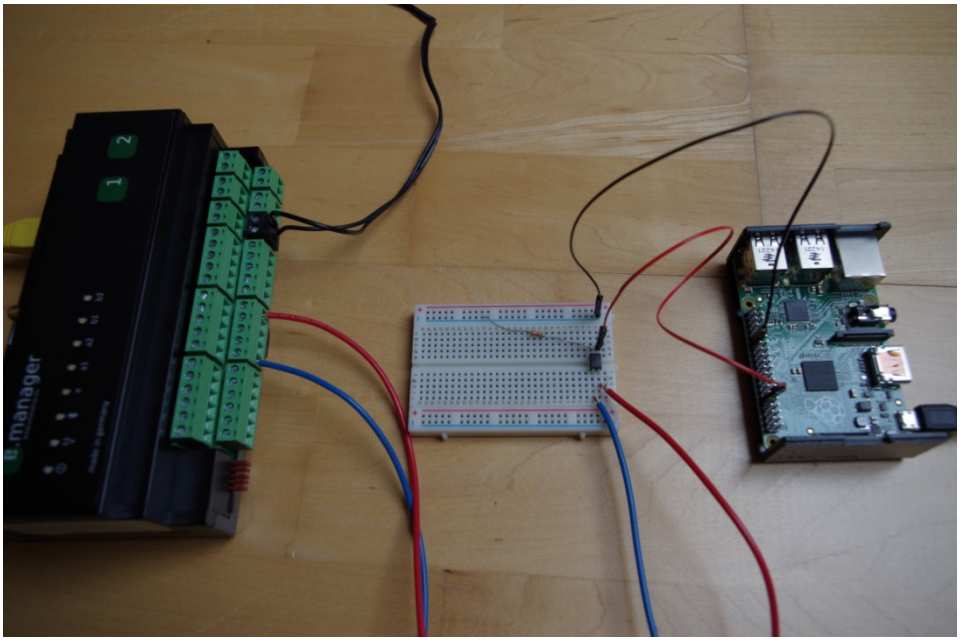

Abbildung 4.2: Aufbau mit e.manager

Die ermittelten Daten des e.managers können über ein Webinterface abgerufen werden. Das Portal von enerserve (http://portal.enerserve.eu) erlaubt es dem Benutzer den Verlauf der einzelnen Geräte zu verfolgen. Neben Übersicht des akutellen Verbrauchs, liefert das Webinterface historische Daten abzufragen. Dabei wird zwischen Tages-, Monats- und Jahresansicht unterschieden. Weiterhin bietet das Interface eine Konfiguration der Anschlüsse.

5 Software

5.1 Modell

5.2 Simulator

5.2.1 Eiswasserspeicher

5.2.2 Steuer-Server

5.3 Steuer-Client

6 Zusammenfassung