Free as in Free Beer

Or, the foundation of civilization

Greg Lehey
LEMIS (SA) Pty Ltd
PO Box 460
Echunga SA 5153
grog@lemis.com
grog@FreeBSD.org
grog@MySQL.com
grog@NetBSD.org

ABSTRACT

Beer is a well-known alcoholic drink, brewed in most countries in the world. It also represents one of the world's most important chemical industries. This paper investigates the central importance of beer in modern civilization, suggesting even that it might have been the reason for mankind becoming sedentary in the first place. It continues with a description of the biochemical processes involved in making beer, and how computers can help.

The foundations of civilization?

Ancient man was nomadic. He hunted animals and ate plant matter such as roots and leaves, and when they were exhausted, he moved on. Starting about 7,000 years ago, that changed: in Mesopotamia, some groups of people stayed where they were.

It doesn't take much thought to realize that maybe the conditions there were so good that they didn't *need* to move on: maybe there were enough animals, roots and leaves there to make moving on unnecessary. Certainly that played a role, but it would not have been enough by itself: the grass is always greener on the other side of the hill, especially for a nomad.

Almost certainly, something developed that was so attractive that people didn't want to move on. But what? Could it have been better food? Possible, but what could be so much better than what people already had?

The obvious thing to do is to look at what people were eating at the time. Grass was high on the list. They didn't eat the leaves, of course: they ate the seeds. The grasses were the ancestor of our modern grains, such as wheat, millet and barley. Like so many foodstuffs, they are seasonal. That's an aspect that hasn't completely changed to the present day, and 100 years ago much food was seasonal. You couldn't get most vegetables for more than a few weeks per year.

So what was special about grass? One thing was that the seeds were durable. They may only have matured over a few weeks per year, but they results would keep not only for the remainder of the year, but if needs be for several years. These issues are the background for the biblical story of how the Israelites came to Egypt.

But grass wasn't the only thing with durable fruit. Pulses (peas, beans and lentils) did so as well. Why didn't they become a staple? They did in China, so there's nothing intrinsic about them that rules them

out. But it was grasses such as wheat and barley that became the staple food of most of Europe and Asia. Only where they weren't available did people choose rice or pulses instead.

We all learnt in school what grass (sorry, corn (sorry, grain)) is good for: making bread. And indeed ancient man made bread. If you've done this yourself, you'll know it's a non-trivial exercise: you have to knead the flour, add yeast, and wait for it to rise before sticking it into an oven. Ancient bread was simpler, of course. They bypassed many of the problems with alternative techniques: instead of yeast, they used lactic acid cultures ("sourdough"), and instead of an oven they heated thin strips of dough on hot surfaces. People continue to do both of these to this day, and the relationship between the Hindi word *nan* and the Spanish word *pan* (both meaning *bread*) suggests that the techniques didn't diverge until relatively recently.

But there's one thing that modern man doesn't have to worry about when making bread: where does the flour come from? Even the word "flour" gives the lie: originally it was "Meal flower", the finest milled meal from the grain.

Grass seeds are not pleasant things to eat. They have a hard husk which even horses prefer not to eat. Inside there's the kernel, but it's pretty hard too (though it doesn't worry horses). You can't make bread out of either.

Primitive man had to do for himself what nowadays specialized industries perform:

- Harvest the grain. This involves going through the fields cutting off the seed-bearing parts of the plant. It's not the worst, but it's still a lot of work.
- Next the seeds need to be separated from the stems. At the same time, the husks are separated from the kernels. This process is almost violent; the name *thrashing* gives an indication.
- After this, and a bit of tidying up, you have the kernels. Wheat kernels look like little white bullets. Chew on them and you're liable to damage your teeth (unless you're a horse). You still don't have anything useful to eat.

Somehow the kernels needed to be made edible. Primitive man found at least two basically different ways to do so:

- Use something hard, usually a stone, to crush them into smaller pieces. The best thing to say about that is that they didn't break your teeth.
- Soften them by boiling. This method is also still in use (porridge, gruel), but it's not nearly as popular as milled flour. There's no reason to believe that primitive man thought any differently: archaeologists report that primitive man spent a lot of time crushing grain.

There must have been myriad experiments to find a middle way. An obvious one would be to soak the grain to soften it, then crush it.

This method turns out to have side effects. The grain is a living organism, capable of growing to a new grass plant. It waits until it gets wet, then it germinates. In the process, enzymes transform the starch in the grain into the sugar needed for the growth of the plant.

The result must have been that many such experiments ended up with a kind of germinated grain soup. It tastes sweet, so there's a good chance that it was popular. But humans aren't the only thing that like sugar: countless micro-organisms, notably lactic acid bacteria and yeast, live off sugar. In the process, they convert the sugar into other byproducts, such as lactic acid and ethanol. These transformations would not have stopped all humans from eating (or drinking) the results. And they would have noticed that, though it tasted sour, it made them feel good. Alcoholic drinks had been invented.

Far-fetched? It's supported by the linguistic evidence: the word *brew* and the word *broth* are related. *Broth* is derived from the old English word *briw*, meaning gruel (a dish of boiled grain). Both are derived from the Old Teutonic word **brû*, as are the German *(ge)bräu* (brew) and *brei* (gruel).

Archaeological evidence shows that man took the production of alcohol seriously from a very early time. But how seriously?

Settling down

Ancient man was nomadic. There were great advantages in being nomadic: it allowed man to be more flexible. Once settled down, man was at the mercy of local conditions. Such problems still occur: if there's a drought, farmers have serious problems. Nomads could wander off and find something else to live off.

So there must have been a corresponding advantage. Bread? It's certainly more difficult to consume than other vegetables, but that in itself makes it unlikely that man would have given up his nomadic life for it. Beer, on the other hand, took time to make, and it offered something that was not readily available by other means. It's very likely that this was one of the main reasons that man started to become sedentary. It's interesting to note that, to this day, nomadic people do not make alcohol.

Not everybody became sedentary at once, of course. Even within a tribe of nomads, it's likely that some people settled down while others carried on their nomadic life. Recent excavations at the site of the ancient city of Caral in Peru (about 2000 BC) suggest that some of the first people to become sedentary were the priests. They left behind ample evidence of use of intoxicants; it's highly possible that beer was considered a holy drink, and the brewers were also the high priests.

The conflict between beer and bread

The fact that beer and bread are made from the same basic ingredient is an issue that still hasn't been resolved. On 23 April 1516, Wilhelm IV and Ludwig X, dukes of Bavaria, proclaimed a decree at Ingolstadt which has since been called a "purity decree" (German "Reinheitsgebot"). In fact, the main purpose of the decree was different:

- To ensure civil prices:

  > From Michaelmas until the Feast of St George, one measure or 'head' of beer will not be sold for more than one Munich penny; and from the Feast of St George until Michaelmas, a measure will not be sold for more than two pennies of the same reckoning, and a head for no more than three heller, under pain of penalty. But when one brews any beer (other than Märzenbier), it will under no circumstances be poured or sold for more than one penny per measure.

  A "measure" ("*Maß*") was 1.069 litres, a little less than an Imperial quart. A "head" ("*Kopf*") was somewhat smaller, but there is no agreement on the exact size.

- To ensure that wheat was kept for making bread, and that only barley (unsuited for breadmaking) was used for beer.

  > Further we decree that henceforth in all our towns, marketplaces and the whole of the countryside, no beer shall contain or be brewed with more ingredients than barley, hops, and water.

  This is the text that is understood to decree the "purity" of the beer. While it certainly did limit the use of herbs and other bittering agents in beer, the important thing was that no wheat was allowed.[1] Conveniently, for brewing barley is an easier grain to process than wheat.

The process

Making beer involves a surprisingly complicated series of biochemical reactions. At the very least it requires:

- Allow the grain to germinate. This germination causes the release of enzymes required for the next step.

---

1. Yes, wheat *is* allowed in German beers now, and it has been for a long time. But it's prohibited by the "purity decree", which hasn't been the legal basis for German beer for a long time. That's the "Vorläufiges Biergesetz" (provisional beer law). But it makes nonsense of claims seen on beer bottles: for example, Schneider writes on the label of their "Aventinus" wheat beer, as sold in Australia: "Brewed according to the Reinheitsgebot".

- Crush the grain.
- Warm the crushed grain in water to allow the enzymes to convert starch into sugar.
- Cool the resultant liquid and allow to ferment. For any reasonable chance of success, yeast needs to be added at this point.

By comparison, making wine requires, at the very minimum:

- Place the grapes into a container which doesn't leak too much.
- Wait. As the grapes shrivel, the yeasts on the skin of the grapes will attack them and cause them to burst. It will then ferment the liquid.

Obviously modern techniques are a little more refined, but even today most wines are made with the natural yeast from the grapes.

The process: more details

The method described in the previous section is the bare minimum. Except for experimental brews such as those described in [Anchor] and [Kirin], modern beers require more complicated versions of each step.

A quick sugar cheat sheet

To follow this description, it helps to understand what sugars are. They include all substances commonly known as *carbohydrates*. The technical name is *saccharide*, and we can distinguish at least four types:

- *Monosaccharides* are the simplest. They consist of rings of 6 carbon atoms with hydrogen and oxygen in the same proportion as in water (thus the name *carbohydrate*). Empirically, they all have the formula $C_6H_{12}O_6$. Common examples are *laevulose* (also known by its old name of *fructose*) and *dextrose* (also known by its old name of *glucose*). Fruit juices, including grapes, contain laevulose. Another monosaccharide is *lactose*, which occurs in milk and which is of interest because it is unfermentable. It is used to make beers with a sweet aftertaste.
- Next come the *disaccharides*, which consist of two monosaccharide molecules joined together. They all have the empirical formula. $C_{12}H_{22}O_{11}$. The most common one is *saccharose*, formerly called *sucrose* and commonly called *sugar*. For brewing purposes, *maltose* is more important.
- A rare *trisaccharide* is *maltotriose*, which is formed during the brewing process. It is of no great importance.
- The *polysaccharides* are what is normally called *starch*. It consists of a large number of molecules joined together in a random pattern. Typically it consists of branches containing straight chains of approximately 100 sugar molecules, depending on the origin of the starch.

The process

- Dry the grain. This may or may not improve the germination characteristics, but it makes it easier to handle.
- Soak the grain in water for about a day, then drain. Barley is the most popular grain, followed by wheat. Occasionally other grains are used, such as rye and oats. The soaking causes the grain to germinate, and this germination causes the release of two enzymes, α-amylase and β-amylase, which are required for the next step.
- Dry the germinated grain in a kiln at temperatures between 40° and 100°. The process of germination and drying is called *malting*, and the resultant grain is called *malt*, independently of what kind of grain it is. Higher temperatures brown the malt, giving a richer flavour.
- Next, the grain is mixed with a significant quantity of water, between 2 and 3 times its own weight, and warmed to temperatures between 30° and 78°. This process is called *mashing*, and the purpose is to convert starch into sugar.

The mash is kept at specific temperatures for periods of time ranging between 10 and 90 minutes (so-called *rests*), enabling temperature-specific reactions to take place:

- At temperatures between 30° and 45°, three changes take place: the pH (alkalinity) of the mash may lower (making it more acid), the grain soaks up moisture (*doughing-in*), ostensibly making it easier for enzymes to react, and β-glucanase can break down some gums. This rest is seldom used with modern malts.
- At round 43°, ferulic acid is produced. With the correct yeast, this forms 4-Vinyl-Guajacol, which has a clove-like aroma much appreciated in Bavarian wheat beers. As a result, this rest is used for such beers.
- The *protein rest* between 45° and 55° allows the enzyme *protease* to break down proteins. This can reduce haze in beer, but it also reduces the head, so it's also not used much.
- Between 55° and 65°, β-amylase is most active. It breaks down the straight parts of the starch into maltose.
- Between 68° and 72°, α-amylase is most active. It breaks down the branches in the starch, allowing remaining β-amylase to convert it further into maltose.

Typically 75% to 80% of the grain is converted into maltose.

- After mashing, the resultant liquid, called *wort*, is separated from the remains of the malt. The English term for this is *sparging*, though the Americans also use the term *lauter*, derived from the German term *läutern*.
- Next, the wort is boiled and hops are added, a process that takes about an hour. The boil has a double purpose: to rid the wort of certain undesirable compounds, including proteins and methyl disulphide, and to isomerize the acids in the hops, bringing them into solution.
- After boiling, the wort is cooled and aerated. The aeration is beneficial to the growth of the yeast in the next stage.
- In the *fermentation* stage, the wort is *pitched* with yeast. This causes numerous reactions, notably the one that converts the sugars into ethanol (alcohol) and carbon dioxide. There are two basic types of yeast, which strongly influence both the progress of the fermentation and the nature of the resultant beer:
  - *Top-fermenting* yeasts form a scum on the surface of the wort during fermentation. Typical top-fermenting yeasts operate at 16° to 20° and complete fermentation in about 3 to 4 days. They are used for ales and wheat beers, and are the predominant British strain.
  - *Bottom-fermenting* yeasts do not form a scum. They also work at much lower temperatures, between 8° and 14°. Due to these lower temperatures, fermentation can take from 7 to 10 days. This style of yeast is used for lager and Pilsner beers, and is the predominant German strain.
- After fermentation, the beer is stored for a certain period of time; bottom fermented beers are stored at a very low temperature, just above freezing, and they can take up to 3 months to be drinkable. All beers benefit from at least a month of storage, however.

Technological aids to beer brewing

Brewing has benefited greatly from technological improvements. Computers are no exception. There are two main areas where computers can help: in calculating the composition of a beer (and thus its character) and in temperature control.

Composition calculations

The following parameters are most important for determining the character of a beer:

- Overall sugar content. This determines the amount of ethanol in a beer. Typical beers contain between 2 and 10% ethanol by volume.
- Overall bitterness. The bitterness is determined by the quantity and nature of the hops, and the length of time for which they were boiled.
- Colour. The colour is determined by the kinds of malt used.

A number of other factors also determine the nature of the beer, but these are both the most important and the most easily quantifiable.

A number of programs are available to calculate the ingredients for making a specific beer style. The best-known calculator is *ProMash*, a commercial program available for Microsoft "Windows". It is also reported to run under *Wine*. It is the de-facto standard for brewing calculations, but from the point of view of this paper it has three disadvantages:

- It doesn't run natively on UNIX and Linux machines.
- It uses American units by default. Even after a relatively painful configuration for metric units, the conversions show the underlying use of American units.
- It costs money.

A number of free programs are also available. None are as comprehensive as *ProMash*.

- *Qbrew* [Qbrew] is a *Qt* application that offers much of the functionality of *ProMash*. Like *ProMash*, it has problems with metric units.
- *Brewsta* [Brewsta] runs under *python*. It's currently in beta test, and I haven't been able to get it to work.
- *Brewnix* [Brewnix] is a more mature package. I haven't tried to use it.
- *brew* [brew] is a collection of utilities that I have written and which I update as needed. They have the advantage of being command-line oriented and thus not requiring mouse usage, which proves to be difficult when your hands are wet or in flour.

Mash process control

In addition to helping plan a beer, computers can help in the brewing process. The most obvious application is temperature control, which is important at different phases of the brew.

The temperatures during the mash determine the characteristics of the beer. In the days before thermometers, there were two approaches to mash temperature control:

- In the British tradition, it was largely a matter of mixing specific proportions of water at room temperatures and boiling water. For example, to mash at 64°, the *strike water*, the water added to the malt, should be at about 70°, depending on the relative amount of strike water and malt. This temperature can be achieved by adding one part of water at room temperature to two parts of cold water (about 10° in Britain). Old British mashes tended to be *single infusion* mashes, where the mash temperature was relatively constant over the duration of the mash.
- In Germany and neighbouring countries, *decoction* was (and is) used. The mash process used in Pilsen in the early 20th century started by mixing the grist with the water and then removing a part of the mash, boiling it and then returning to the mash. By the choice of the quantity to decoct, multiple rests can be accommodated.

Mash temperature control is complicated, and it's an ideal application for computers. Unfortunately, to my knowledge no freely available software exists to perform the temperature control, though both *ProMash* and *brew* offer aids to calculate strike water temperature and decoction quantities.

Fermentation temperature control

As mentioned above, the temperature of fermentation is important for the quality of the final beer. Commercial breweries maintain the temperature of fermentation vessels with an ethylene glycol jacket around the fermentation vessels. Home brewers in Australia tend to use polypropylene fermenters with volumes between 20 and 33 litres, which are still relatively easy to move when full. They need some space for froth, so the typical batch in a 33 litre fermenter is between 20 and 25 litres. For some reason, the old "5 gallon" batch dies hard, and many choose a volume of 23 litres.

For this kind of batch, a glycol jacket is both very expensive and probably inappropriate. I have never heard of one being made. The straightforward way to maintain temperature is to put the fermenter into a refrigerator.

Under these circumstances, it's relatively trivial to add a thermostat to maintain the temperature at the desired level. With the right choice of thermostat, it's also possible to arrange to warm the fermenter when the outside temperature drops below the desired fermentation temperature. [old-control] describes a method using old fridge (which itself has a defective thermostat) with an off-the-shelf digital thermostat located on the lid of one of the fermenters. The thermostat is set to turn on the fridge compressor when the surrounding temperature is too high, and the lamp located beneath the fermenters when the surrounding temperature is too low.

This approach is simple, cheap, straightforward and ineffective. The problems are:

- The thermostat switches mains voltage, which is a safety hazard. To meet safety requirements, a couple of external relays and a low-voltage power supplies are needed.
- The inside of the fridge can be humid. The thermostat is not suited to humid environments, but the temperature sensor is not removable, so it must be placed where the temperature is to be measured. Further experiments showed that the humidity problem could be combatted by placing the thermostat in a thin plastic bag.
- The thermostat is a simple switch-over model: one of the contacts is always closed. This means that the device is always either heating or cooling. In practice, it's possible to disconnect the output to the cooler or heater, but this results in less accuracy when the outside temperature is close to the set temperature.
- The temperature is measured in the wrong place. The temperature on the lid of the fermenter is of only marginal interest. Fermentation generates heat, and so the wort temperature will be higher than the surroundings, as much as 6° higher than the set temperature. This is inadequate for the purpose.

The temperature problem is even more complicated than it looks. The fermenter is only cooled from the outside, so the temperature in the middle will be higher than on the outside. Where is the correct place to measure the temperature?

There are obvious places to measure the temperature: the surface of the fermenter and the middle of the wort. The former will measure too low a temperature, and the latter one too high a temperature. A compromise would be to put the sensor about halfway between the axis of the cylinder and the surface, which comes closer to an "average" temperature.

Measurements inside the wort have two basic problems:

1. Obviously the cables need to be encapsulated in something to protect them from the wort, and even so it's difficult to maintain good sanitary conditions with cables stuck into the wort.

2. The wort reacts very slowly to external temperature changes; a step change of 10°, typical when refrigerating, can take hours to become noticeable. This gives rise to the danger of overcooling, followed by overheating.

Placing the sensor on the surface of the fermenter has neither of these problems. It's true that during cooling and heating it results in a slight offset from the wort temperature, but this is not a problem once the wort has reached a steady state.

A usable solution would thus measure the temperature of the outside surface of the fermenter and control a heater or cooler, with some temperature tolerance during which the device neither heats nor cools; ideally the system would spend most of its time in this condition. [Sorenson] describes such a device based on purely mechanical components. The web page does not describe the accuracy of the temperature control, but it's reasonable to assume that it's adequate.

The problem with Sorenson's approach is the components. As the article describes, they are no longer available. It is also not suited for more complicated procedures, such as slow, constant temperature changes such as are used after primary fermentation: bottom fermenting beers frequently need a *diacetyl rest*, a raise of temperatures to about 16° (see below for an example), and top fermenting beers benefit from a cooler secondary fermentation. In each case, a step change in temperature is not desirable.

Based on this background, I created a more flexible solution, based on an little-used Intel 80486 computer with a relay board and a digital thermometer input.

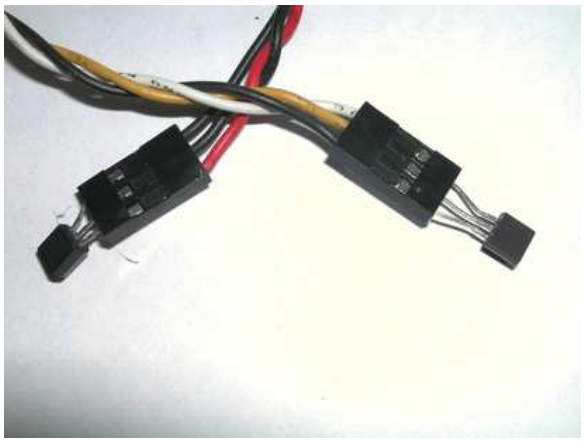
Overview

The control system uses a computer to monitor a number of temperatures inside a fridge and turn on either the fridge motor to cool the surroundings, or a light bulb to warm them:

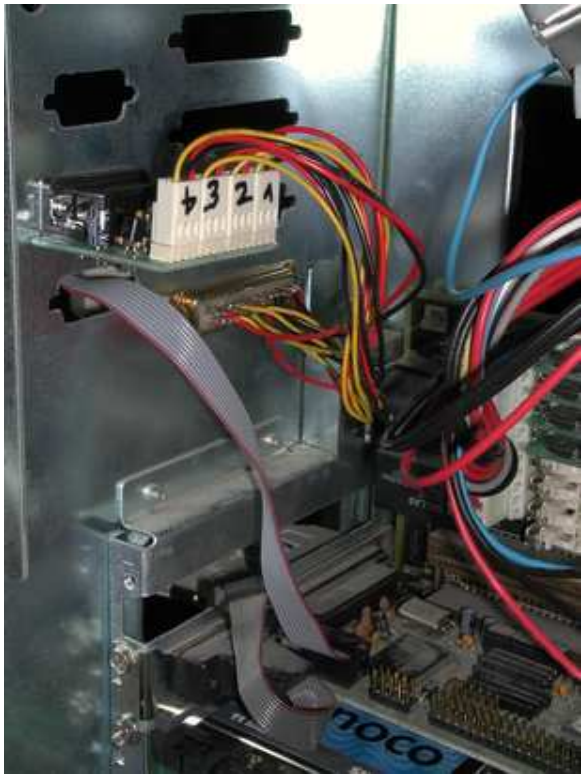
The equipment I used is:

- An old computer, of course. The size of the one I chose is simple: that's what I had lying around. It's an Intel 486-DX2/66 with 16 MB of RAM, something you can probably pick up for free if you know where to look. It's running FreeBSD (*http://www.FreeBSD.org/*), of course.
- A temperature logger kit (*http://ozitronics.com/kitlist.html#k145*) available from from Ozitronics kits (*http://ozitronics.com/*). It connects to the system via the serial port.
- A relay board (*http://ozitronics.com/kitlist.html#k74*) also available from from Ozitronics kits (*http://ozitronics.com/*). It connects to the parallel port and controls up to 8 relays with up to 250 VAC and 10 A, though they recommend additional wiring for currents of over 5 A. A fridge typically uses a maximum of 3A, so this is of academic interest only.

The real fun in getting this working wasn't the hardware, which is easy enough to get. It's also not really the software, which I wrote myself, and which I'm still tweaking. The real problem were the little details and connectors and things. I spent a lot of time trying to decide how to connect things. Finally I discovered an old computer lying around without a mother board, so all the front panel connectors were hanging loose. That's exactly what I was looking for to mount the temperature sensors:

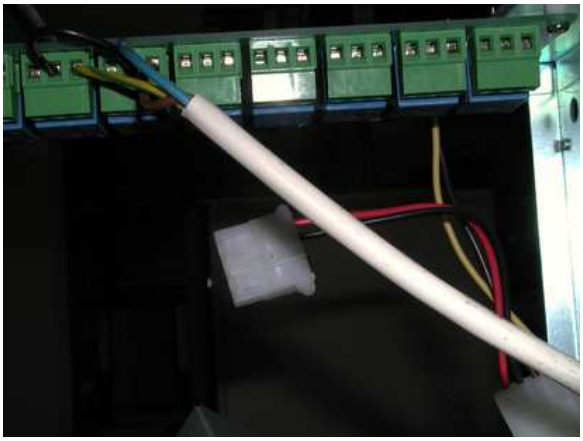


Other issues in the mounting included the fact that the kits are designed for external mounting (and the relay board needs a 12V power supply). I wanted to mount both inside, which had the added advantage that I could use the computer power supply to power the relay board. The problem was a certain amount of external cabling:

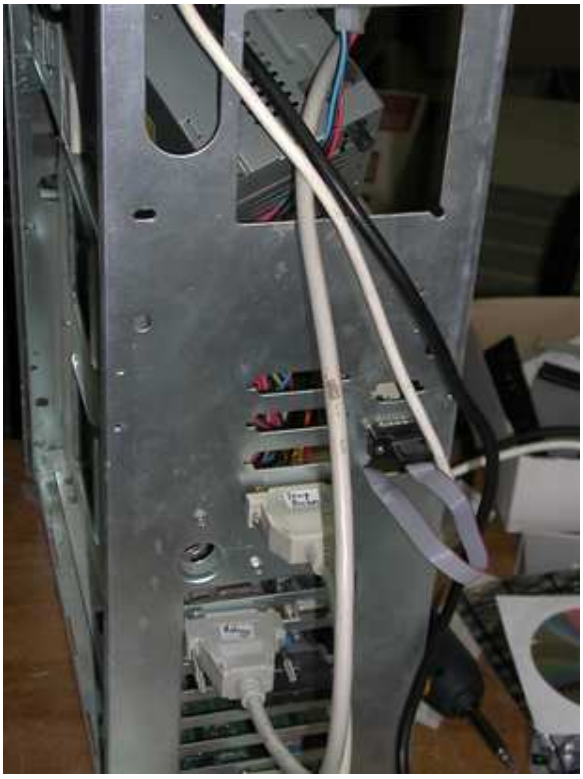


This one shows the temperature probe assembly. There are no mounting holes on the probe board, so I had to mount it by its 9 pin serial connector. I had already connected to probe cables to a 25 pin connector. I wanted it inside the case, so I had to connect the flat cable to the serial port on the outside of the case (the grey cable going out through another cutout just below the probe board). I need to find

some kind of plate that I can use to mount it inside the case.



This one shows the relay board with the mains power connections.



A view of the back of the computer. This shows a number of things:

- The lower cable goes from the parallel port back inside to the relays. It would be nice to have internal cabling, but I don't know of any parallel ports that connect to a header on the board. They're all connected directly to an external connector.
- Above that is the temperature probe cable, as shown before.
- Higher and to the right, the flat band cable mentioned previously.
- At the top are the relay power outputs (white) and the computer power cable (black). The power supply is in at an angle because it was originally designed for a smaller case, and the internal cables are too short to allow normal mounting. It has since been replaced.

Installation

The next step was installation in the laundry:

The temperature sensors can be seen in the enlargement on the web site.

- The external ("room") sensor is on the outside left of the fridge.
- The internal ("ambient") temperature sensor is in front of the 25 pin connector. It's fastened to one of the bars of the grille.
- The wort temperature sensor is taped to the outside of the fermenter. It's covered with some bubble foil to minimize the effects of the ambient air.
- The fourth sensor is for a second fermenter. In this image it's unused and hanging down in front of the light.

The software

The software reads its parameters from a configuration file which can define a large number of parameters. Almost all of them have defaults. The most important parameter to change is the fermentation temperature. Two temperature parameters are provided to handle temperature ramps:

- `starttemp` is the temperature to set when the program starts. If the temperature is to be constant, this is all that needs to be set.
- `endtemp` gives both a temperature and a time for the end of the ramp. For example, to start at 19° and drop to 14°, the file might contain:

```
starttemp 19
endtemp 15 15 september 2005 0:0
```

  This would start out at 19° and progressively cool to 15° by midnight between 15 and 16 September. This choice of parameter is an inadequate approach to the issue of maintaining the ramp even if the program is stopped and restarted during the ramp. More effort needs to be addressed to this issue.

Cooling and heating

The simplistic approach to temperature control is shown by the digital thermostat described above: if it's too cool, heat. If it's too warm, cool. This is clearly inadequate. At the very least, the system needs to accept a minimum range in which it will neither heat nor cool. Without such a range, it would be continually either heating or cooling.

In fact, there are a number of issues here:

- If the idle temperature range is too wide, the accuracy of temperature control suffers.
- If the idle temperature range is too narrow, the system can end up alternating between heating and cooling. This, too, diminishes the accuracy of the temperature control.
- If the cooler is turned on or off too frequently, it can cause damage to the system. Typical refrigeration units have safeguards to ensure that they don't turn on or off more than about every 60 to 300 seconds.
- When the heater or cooler turn off, the air temperature differential ensures that the fermenter surface continues to increase or decrease in temperature: it overshoots the mark.

The temperature control system attempts to address these issues, with varying degrees of success:

- The original intention was to keep the temperature within 0.5° of the goal temperature. As controlled by the PIC, the sensors have a resolution of approximately 0.07°, so this seemed reasonable. In fact, it has proven to be possible to maintain the temperature to within 0.10° of the goal temperature, a very satisfactory result.
- The program should learn from its mistakes and compensate. For example, it should learn to what extent the temperature overshoots, and how long it takes to recover, and choose its parameters accordingly. This aspect is still not satisfactory, and in the meantime a number of tuning parameters are available to tune the system.

Sample output

The software produces output in a number of formats:

- It outputs temperature statistics approximately once per second onto the controlling tty:

```
Time      Brew   Probe  Base  Ambient Goal  Offset  Room
            61       2
14:49:41  18.31  15.25  18.31  15.68  18.21   0.10  17.87
Status: Cooling
```

  In this example, only one of the two probes is being used to actively control the temperature. The information is:

  - `Brew 61` is a label specified in the configuration file to help identify the brew:

```
fermenter1label  Brew-61
fermenter1label1  Brew
fermenter1label2  61
fermenter2label  Probe 2
fermenter2label1  Probe
fermenter2label2  2
```

  - `Probe 2` is the other temperature probe. In this example, it is not connected to anything. It is still functional, so it shows a value roughly corresponding to the ambient temperature probe. The difference between the two is not an issue of the accuracy of the measurement: it

shows the natural temperature differences in different places in the fridge.

- `Base` is the calculated base temperature that is being controlled. It is somewhere between the temperature of each of the two fermentation temperature probes. In this case, since only one brew is being made, it is set to be identical to the temperature of brew 61:

  ```
  probe2factor 0
  ```

  `probe2factor` is a (floating point) value between 0 and 1 which specifies the extent to which the temperature of probe 2 contributes to the calculation of `base`. The formula is:

  ```
  basetemp = temps [fermenterprobe] * (1 - probe2factor)
    + temps [fermenter2probe] * probe2factor;
  ```

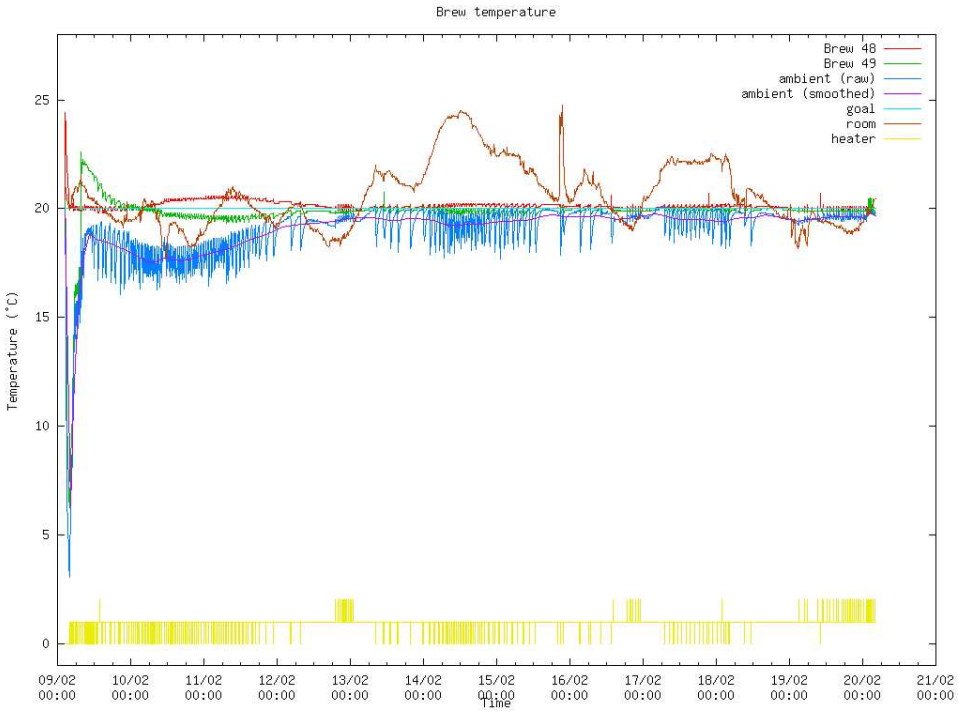
- `Ambient` is the air temperature inside the fridge.
- `Offset` is the difference in temperature between the `Goal` and `Base` temperatures.
- `Room` is the temperature outside the fridge.

The bottom line shows the current status; it is cooling the wort to bring the temperature down from 18.31° to the current goal of 18.21°.

- This representation is also available across the Internet with the command

```
$ telnet brewer.lemis.com 4135
Trying 192.109.197.147...
Connected to brewer.lemis.com.
Escape character is '^]'.
Time      Brew    Brew  Base  Ambient Goal  Offset  Room
43      44
18:23:39  19.06  18.93  18.99  18.12  19.00  -0.01  23.31
Status: Idle
Connection closed by foreign host.
```

- The other representation is a graph. A recent graph (typically less than a day old) is shown on my brewing home page (*http://www.lemis.com/grog/brewing/*). A typical graph looks like this:[1]



1. Y axis: temperature.
2. X axis: Time and date.
3. Red line: temperature of first brew (Brew 48 in the example). It starts off at nearly 25° and quickly drops to 20°. After about a day it gradually rises to about 21°, and takes about 2 days to return to the set temperature.
4. Green line: temperature of second brew (Brew 49 in the example). It starts off a little later than brew 48 at about 23°, and also quickly drops to the set temperature. At the same time that the temperature of brew 48 rises to 21°, it performs a complementary drop to about 19°. This is one of the drawbacks of the system: brew 48 is fermenting vigorously and generating significant warmth. At this point, brew 49 is not fermenting as vigorously. Since the heating or cooling is only by the ambient air, it is not possible to get exact temperature control for both brews. The system controls the base temperature, not shown on this graph, which remains close to 20°.
5. Dark blue line: ambient temperature in the fridge. This is the temperature of the air that cools or warms the beer. Initially it drops sharply to about 3° to cool the beer down to the set temperature; then it shows a very jagged curve as the cooler cuts in and out.
6. Purple line: a smoothed version of the ambient temperature. The jagged nature of the ambient temperature curve makes it difficult to see the average.

   It's worth noting that throughout the fermentation, the ambient temperature is lower than the wort temperature, due to the need to remove the heat of fermentation.

1. Note that the conference proceedings are printed in black and white, which makes it very difficult to read this example. Colour versions are available in the slides and the original of this paper; see the bibliography for details.

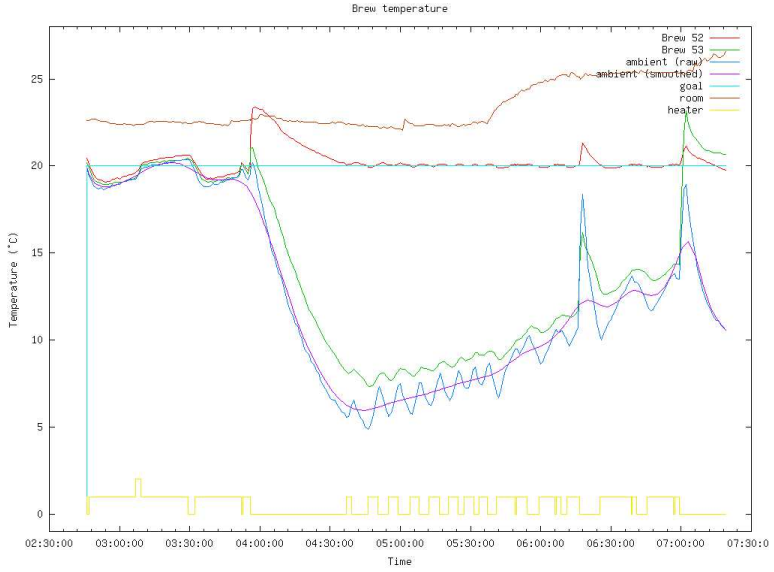
7. Cyan line: goal, the temperature that the wort should achieve. In this case, it's a constant 20° throughout the 12 day fermentation time.

8. Brown line: room: The temperature outside the fridge. This is purely informational; it (currently) isn't processed in any way. As can be seen, it has almost no effect on the ambient temperature or wort temperatures.

9. Yellow line: indicates whether the system heats (line up from centre) or cools (line down from centre). There is a certain correlation here with the outside temperature, of course.

   I have been experimenting with a further line showing the smoothed rate of heating or cooling, but this has not been completely satisfactory. Some of the other graphs show the effects.

The following graph shows the first five hours of two fermentations in more detail, from the time the system is started until shortly after the second brew is placed into the fridge:



- At about 2:45 (UTC in this example) the system is turned on. The sensors are not connected, so the system sets the temperature to 20°. In this state temperature control is poor, since there is little thermal mass to stabilize the temperature.

- At about 3:55 UTC, the first brew (52) is placed in the fridge at a temperature of about 23°. The system immediately starts cooling and continues until about 4:35, by which time the outer surface of the fermenter has reached 20°. At this point the ambient temperature is about 6°.

- The wort is still warmer, and to maintain the outside temperature the system has to continue to cool at regular intervals, gradually raising the ambient temperature.

- At 6:15 UTC, the door of the fridge is opened, causing a sharp rise in the ambient temperature. The cooler turns on again and cools back to the necessary ambient temperature.

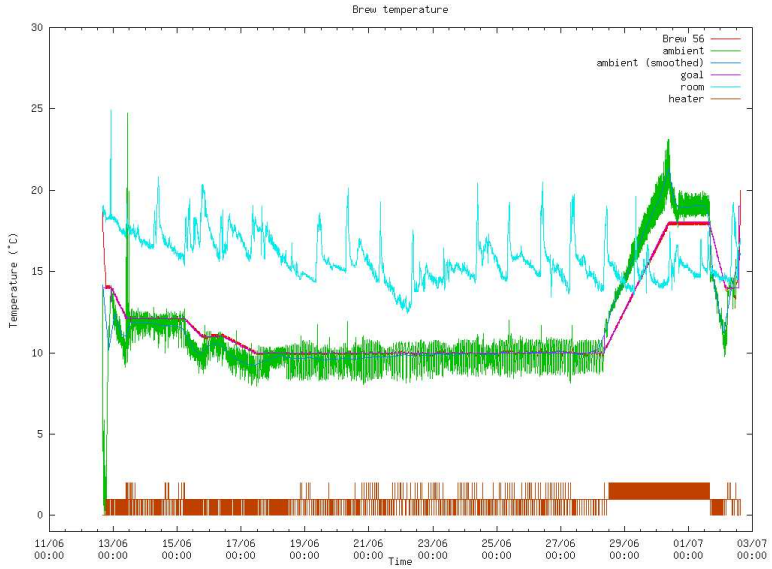
  In this example, the temperature of brew 52 also goes up by about 1°. This is indicative of a poor contact between the sensor and the fermenter: the ambient temperature influences the reading more than it should.

- Finally, at about 7:00 UTC, the second fermenter is placed in the fridge and connected. Again the ambient temperature and the temperature of brew 52 rise, but more notably the temperature sensor for brew 53 (green line), which previously has been tracking the ambient temperature, rises as well. Again the system cools back to the set temperature.

It's worth noting here that during this time the base temperature should be set to the temperature of the first fermenter (`probe2factor` set to 0), so that the low reading for the second temperature sensor

doesn't influence the temperature. After adding the second fermenter, it should be set to 0.5.

The next graph shows a more complicated set of temperature transitions:



This graph shows the brewing of a bottom-fermenting beer. Fermentation starts at 14°, but as soon as it is under way, the temperature is lowered first to 12°, then to 11° and finally 10° for the bulk of the fermentation. After two weeks, the temperature is ramped to 18° for a diacetyl rest, then dropped again to 14° until bottling.

This graph also shows a weakness of the system: between 19 June and 27 June, the heater turned on several times to compensate for overcooling. It's possible to get rid of this problem by lengthening the hold-off time between turning the cooler off and turning the heater on, but ideally the system should learn this itself and not require external intervention.

Current status

The first version of the software had the typical "open source" disease: the documentation is pretty primitive. Since release I've found some minor bugs which I have mainly fixed. When the remaining bugs are fixed, I'll write some proper documentation.

Experience with the system

The system has now been in use for about 14 months. It has done its job well, and the accuracy of the temperature has surpassed all expectations. It normally maintains the temperature to within 0.1°, a surprisingly good result considering that the resolution of the sensors is about 0.07°. There have been some issues, however:

- The connections to the temperature sensors have been less than ideal. The choice of connectors is not well-suited to the environment, and the connections can corrode.
- The PIC microcontroller does not handle misbehaving sensors well, and can send nonsensical values to the computer.

  I have upgraded the software to ignore obviously invalid data records from the PIC microcontroller, but it is still possible to get valid-looking data records that are just incorrect. Some heuristics may help here, but so will an alternative solution: the temperature sensors can be connected directly to a serial bus, and with some level transformation logic and a modified serial line driver, a theoretically

infinite number can be handled by the main computer. In this case, more sophisticated error handling should be possible.

- Attaching the temperature sensors to the fermenter is more complicated than it looks: the accuracy of the temperature control depends very much on the good contact between the sensor and the surface of the fermenter. If there is even a small gap, the accuracy of the control diminishes greatly, as shown in the second graph.
- Currently the software runs as a single process and thread. The log files are NFS mounted, so if there is a network problem, the entire process can hang. I intend to handle this problem in two ways:
  - Use a separate thread to handle output where needed.
  - Migrate to a system where the remote computer requests information. This is partially provided by the TCP connection, but it is obviously too much overhead to set up a new connection on every enquiry (which typically is once a second). I'm planning a UDP alternative.

Future directions

There's still plenty of work to do on the project, which I'll do when the itch gets strong enough. Obvious ones are:

- Write proper documentation.
- The system isn't protected against power failures (there's not much point, given that fridges aren't either). But if it fails during a temperature ramp operation, there's no way for the system to know what temperature to set when it restarts. Saving information to disk should help there.
- Modify the system to cool multiple fridges. Even the base hardware can handle four temperature sensors, enough for two fridges with one fermenter in each, and the relay board has eight relays, enough for four fridges. With an additional temperature sensor board, it could easily control three fridges. The real issue is in the program structure: how to define the parameters for each fridge, and how to keep track of them? That's a soluble problem.
- The system does not do a very good job of learning when to turn the power on and off. This issue requires further consideration.
- As mentioned above, a UDP interface would make it easier to perform repetitive queries.
- The current system only addresses fermentation temperature control. As mentioned, it would also be desirable to have a system for controlling mash temperature. Plans are under way to do this.
- Beer is not the only thing that can benefit from temperature control. Australian domestic ducted air conditioning systems have appallingly bad temperature control; a modification of this program could handle that as well. Many other such applications exist.

Bibliography

[Anchor] *Sumerian Beer Project*, published by the Anchor Brewing Company. *http://www.anchorbrewing.com/beers/ninkasi.htm*

[Biergesetz] Bundesgesetzblatt 1993 Teil I Seite 1400, *Vorläufiges Biergesetz*. The current German beer law. *http://www.jura.uni-sb.de/BGBl/TEIL1/1993/19931400.1.HTML*

[brew] Brewing toolkit. Sources are at *http://www.lemis.com/grog/brewing/src/brewing*, but there is no documentation beyond the sources.

[Brewnix] Open Source Brewing Software. *http://brewnix.sourceforge.net/*.

[Brewsta] Open Source Brewing Software. *http://sourceforge.net/projects/brewsta/*.

[Helfferich] *Beer Before Bread*, by Carla Helfferich, in the Alaska Science Forum. *http://www.gi.alaska.edu/ScienceForum/ASF10/1039.html*

[Kavanagh] *Archaeological Parameters for the Beginnings of Beer*, Thomas W. Kavanagh, Ph.D. Published in Brewing Techniques September/October 1994. *http://www.brewingtechniques.com/library/backissues/issue2.5/kavanagh.html*

[Kirin] *New Hypothesis Validated on Method of Brewing Beer in Ancient Egypt*, article published by Kirin brewery without author attribution. *http://www.kirin.co.jp/english/ir/news_release020802.html*

[old-control] A description of a first attempt at what led to the temperature control system. *http://www.lemis.com/grog/brewing/old-temperature-control.html*.

[Reinheitsgebot] Wilhelm IV of Bavaria, *Das Reinheitsgebot von 1516*. The original (Bavarian) text of the "purity decree". *http://mitglied.lycos.de/bieronline/reinheitsgebot02.htm*

[Sorenson] *Temperature Control and Refrigeration*, by Paul Sorenson. A description of a mechanical temperature control system. *http://www.brewiki.org/wiki/homebrew/moin.cgi/TemperatureControl*.

[Tempcontrol] *Brewing Temperature Control System* , by Greg Lehey. The subject of this paper. *http://www.lemis.com/grog/brewing/tempcontrol.tar.gz*.

The original paper (including colour images) and slides for this talk are available at *http://www.lemis.com/grog/Papers/freebeer/*.