# Modelling Neuronal Spike Codes

Gerold Lee Gorman

Contemporary approaches to artificial intelligence seem to be based upon one of two main strategies. One method involves tensor flow algorithms which are typically used to try to find cascades of matrix multiplications that are clamped with sigmoid functions. Then there are the language models that are based on Markov chain-based prediction strategies. Thus, I propose a third approach that uses principles derived from sigma-delta modulation techniques to carry out the mathematical operations that are associated with any neuronal topology. As this project is developed, there will be opportunities to discuss applications such as feature recognition, data compression, chaos theory, and so on, along with possible applications such as creating new approaches to the development of learning models.

## Description

Now you are probably wondering, just how could a new approach to "*modeling neuronal spike codes*" be used as a piece of assistive tech. Well, there are several applications that could prove not only relevant but might also have radical implications for how we live our daily lives, just as other technologies such as mass communications, personal computing, the Internet, and smartphones have transformed our lives. Recent reports have suggested that GPT-4 right now could do about 50% of the work that is presently being performed by something like 19% of the workforce. This is clearly going to have a major impact on how people live and work from here on out.

Then again, A.I. is already having major effects on many people's activities of daily living. I routinely ask O.K. Google what time it is, or to set an alarm, or what the latest news is about some subject of interest, whether it is the latest dirt on Trump, or the NFL scores, or the weather, for that matter. As the quality of the interactive experience improves, it remains to be seen just how many people will automate more and more of their lives.

Yet what GPT-4 is doing is obviously going to have much more profound implications, even with all of its present faults. One of which is the fact that the training set used by Open-AI is not open source, so we don't know if they have tried to incorporate every Biden speech ever given, every Trump speech ever given, the text of every decision ever handed down by the Supreme Court, as well as whatever else might be available, such as other obvious things, like the complete works of Shakespeare, the Adventures of Tom Sawyer, and so on. So we don't know off hand what political, cognitive, ethical, or even factual biases are being programmed into it, and this is a potential disaster.

Likewise, there are media reports that suggest that it took something like 3 million dollars worth of electricity to create the model for GPT-3, so obviously this isn't exactly a solo adventure that the casual hacker can undertake, let's say by downloading everything that can be possibly downloaded over of good fast Internet connection, like if you have Gigabit fiber to your home, and can rsync Wikipedia, in addition to whatever archive.org will let you grab, while you still can.

There must therefore exist someplace in the legitimate hacker ecosystem for some kind of distributed AI-based technology wherein it might become possible to create an AI that is continuously improved upon by some kind of peer-to-peer distributed *mining* application.

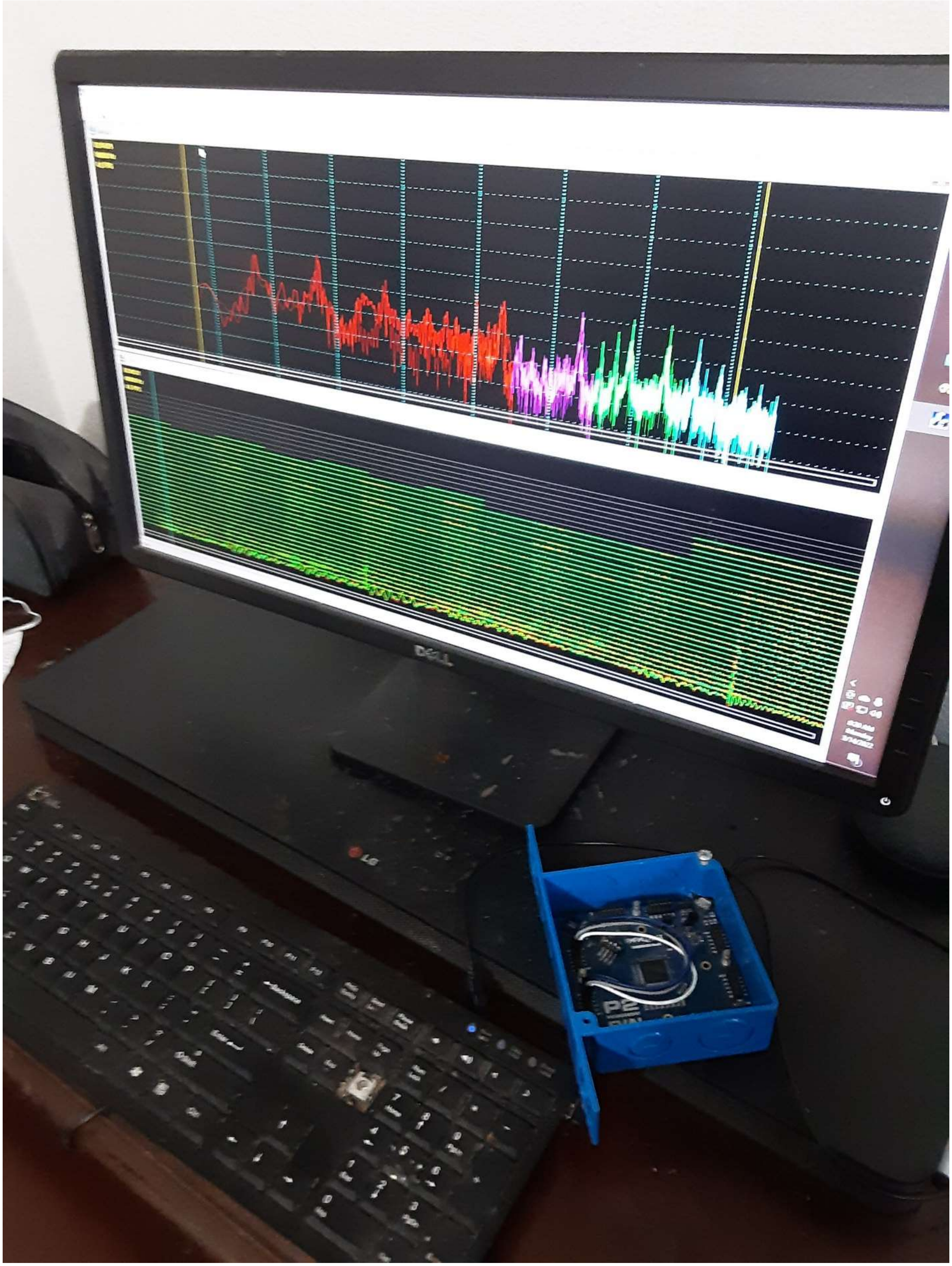
There, I said it - what if we could mine structures of neuronal clusters, in a manner similar to existing blockchain technologies? Off the top of my head, I am not sure just what the miner's reward would be, and maybe there shouldn't be any, at least not in the sense of generating any kind of tokens, whether fungible or not, especially when we consider the implications with some of the present crypto mess.

Yet clearly something different is needed, like a Wikipedia 2.0, a GitHub 2.0, or a Quora 2.0, or whatever the next generation of such an experiment might turn out to look like. So other than awarding prestige points for articles liked, I am not sure just how this might play ut. Perhaps by generating memes, that survive peer review, contributors might gain special voting rights, such as the authority to allocate system CPU compute blocks, like if someone has a protein folding problem that they are interested in, instead of simply reviewing existing magazine articles digests, and suggesting improvements.

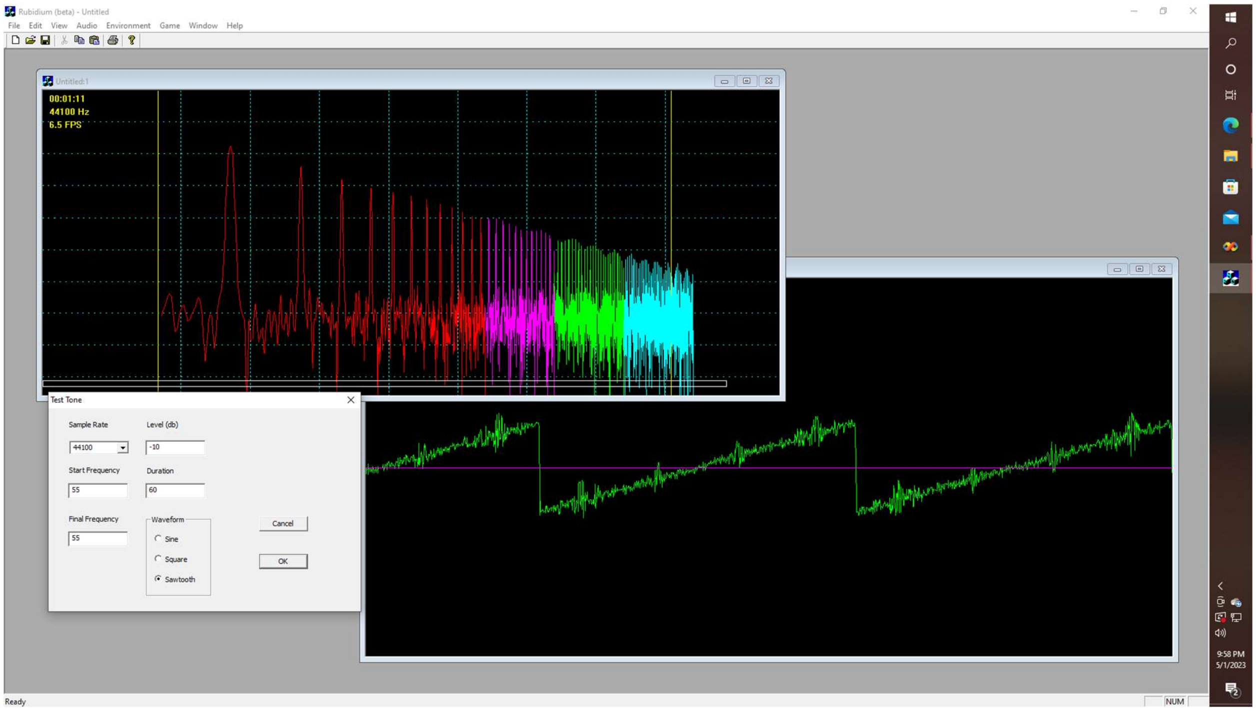
Now this gets way ahead of things of course. First, we need to develop a new approach to AI that just might turn out to be an order of magnitude more efficient, from the point of view of electricity consumed, for example, than the contemporary tensor-flow-based methods, and which might have the ability to solve some of the types of problems that typically are thought of as requiring some kind of quantum computer. Even if we can't produce the same level of performance that a true quantum computer might provide, we should nonetheless be able to produce some interesting results by tinkering with simulated asynchronous analog neurons in a digital environment, hopefully by starting small, and then figuring out how this might scale up.

So let's try modeling neuronal spike codes by using sigma-delta modulation techniques. If nothing else comes of this, maybe we can come up with a new and improved noise generator that will make the sounds of rain, wind, or waves crashing on the beach. Then who really knows? Maybe a new and improved white noise generator is what some people need in order to get to sleep at night! Right? That approach could bear fruit in other areas, however. Hence, some applications that are currently being performed with the traditional approaches to AI just might benefit if we could incorporate a new approach to digital signal processing (DSP) that takes advantage of whatever newly discovered neuronal topologies that we might come up with.

Thus, just in case implementing SkyNet seems a bit far off, or even grandiose, we should at least try to consider how some of the traditional applications, like pattern and speech recognition, and so on just might benefit from the new algorithms.



Or a sawtooth.  O.K.  I have this application where I was doing a simulation of how a sigma-delta analog to digital converter works a while back, and this is what I came up with when I used a VERY low bit rate, like 32 bits per sample bitstream model that in turn was being fed through a Gaussian window function in order to recreate 16-bit WAV stereo audio from the 1-bit data stream, with the obvious question being, therefore, just how low can we go, and still get something that might be acceptable in some context, like speech recognition, or music transcription, that is, if all we need to do is recognize pitch, or detect when a particular vowel is being spoken, etc.



00:01:11
44100 Hz
6.5 FPS

Sample Rate: 44100
Level (db): -10
Start Frequency: 55
Duration: 60
Final Frequency: 55
Waveform:
○ Sine
○ Square
● Sawtooth

And of course, it looks noisy, like what we see in a science fiction TV show like Lost-In-Space where John Robinson speculates that some derelict alien spacecraft has a "crystalline power source", or maybe Fantastic Voyage, where some scientists have to try to perform some brain surgery on another scientist by boarding a submarine that is then shrunk down to the size of a red blood cell.

But wait a minute!  Nowadays, we have fractal coding, chaos theory, and some notional constructs that establish some kind of covariance between the so-called "fractal dimension", Feigenbaum's constant, and a bunch of other stuff that more rigorously generalizes the theory that there is some kind of relationship between the "logistic map" and the otherwise very broad field of "convolutional codes", and of course if sigma-delta is a type of convolutional code, they why not build something based on Chua's circuit that actually tries to model a neuron or two, and see if we can trick an otherwise preposterously simple circuit into doing something like pitch or even vowel recognition, that is by using a hybrid approach that incorporates op-amps, flip-flops, and possibly a 555 or two?

I uploaded a background image for this project that I found on Wikipedia.  The original source can be found at https://en.wikipedia.org/wiki/File:Delta-Sigma_schematic.svg, which also contains the copyright information which states that that particular image is made available under a Creative Commons License.  For use with this project, I converted the file to jpeg and cropped the image to better fit with the way that Hackaday seems to want to further scale and crop whatever you upload for display purposes.

Later on, I will be creating a new project on GitHub entitled Code Renaissance which I am thinking will become a repository for some classic AI stuff, as well as other material that might be useful.  Thus one approach that I might want to try is to obtain the source code for the classic chatbot MEGAHAL, and then modify it to make use of the modified Tensor flow style evaluator which I have described so far, i.e., by attempting to use sigma-delta modulation as an "intentionally noisy" ALU that can be run with reasonable efficiency on any microcontroller.

Now as it turns out, the Parallax Propeller P2 chip might be ideal here, since it is capable of performing sigma-delta analog to digital conversion on all 64 pins, which seems to suggest that implementing some kind of simple neural network might just be a matter of choosing some pins to act as outputs, and some to act as inputs, and then tie a group of outputs together via resistors, and let them feed into some of the other pins which could be used as inputs.

Then we can attempt to implement some of the classic neuronal topologies, like Hopfield, feed-forward, back-propagation, adaptive resonance, etc., in addition to the contemporary transformer model. How that might fit into creating a Chat-Bot which can act as a personal assistant or provide some "simulated companionship" remains to be seen, but nonetheless, such a project seems very doable.

## Beyond The Uncanny Valley of the Shadows

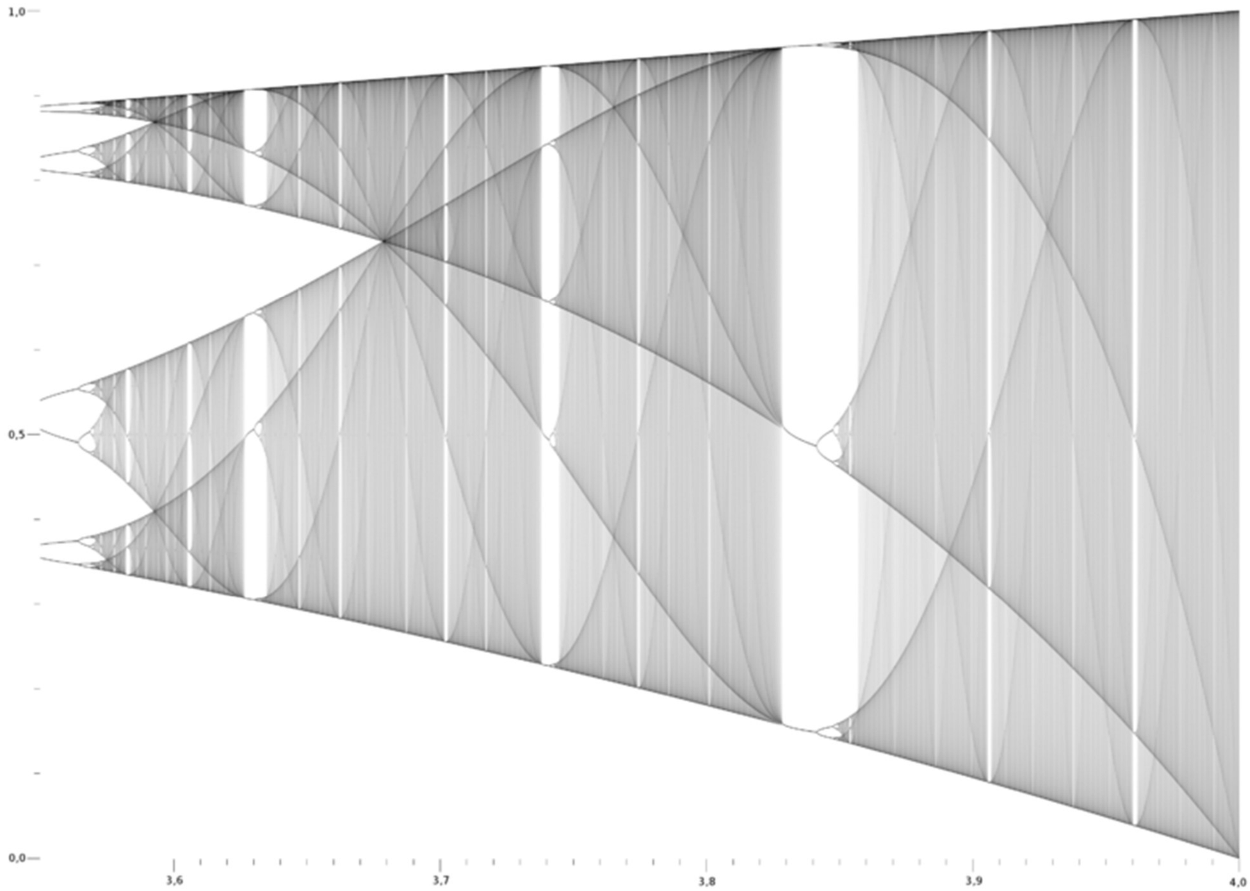
I was thinking about writing a log entry with the title "Avoiding Gestalt Conflicts", and that one would have begun with some repeat of a previous rant about how "if nothing is better than steak and eggs for breakfast" and "if stale cereal is better than nothing" then does it not follow that if A is better than B, and if B is better than C, then A must also be better than C also. Then besides that, where are Alice, Tweedle-Dum, and Tweedle-Dee when we need them?

Well, you get the idea, or do you? If you know what Galois fields are, then you might, in addition to the notional concepts of "frame" and context", you should also be able to contemplate some concept of "circular dominance", and based on what I think I remember about those little neon flasher kits that Radio-Shack sold when I was a kid, I think that one of the things that "random connections of neuronal circuits" might end up doing, so to speak - is that they might somehow "adventitiously" create ad hoc ring oscillators, which in turn might give rise to spontaneous Hopfield Networks.

Now, let's suppose that when such networks fire off in random patterns, so to speak, for a long enough period of time, maybe they do something akin to the accumulation of lactic acid in muscle, for example. In effect, they might experience fatigue, so that perhaps such a network will oscillate for a while, and then it might shut down because it could possibly become inhibited by another network that then comes online. If that circuit then fatigues, then what? Does the first circuit then come back online, now that it has had some time to get some "rest and refresh"?

I am imagining a classic double scroll oscillator here, of the type that can be modeled with Chua's circuit. Yet I want to try implementing it using a neuronal topology as stated earlier.

Alright, I found a public domain graphic of the chaotic region standard logistic map on Wikipedia, and for those familiar, and unfamiliar - it looks like this:



And in other news, one of my favorite pieces to practice on the piano is an "easy" version of the Animals - House of the Rising Sun. So for a good time, if you have access to an electronic keyboard try the pattern 1-2-3-5-4-3 in the right hand, at 120 BPM, 3/4 time in the key of A-minor. Then try to add the sequence A-C-D-F-A-E-A-E in the left hand with the A being the top note and then going down to the C, and so on. So the left thumb is on an A and the little finger is on a C. Try it. Five minutes to learn, a lifetime to master. One full note per measure in the left hand, so that the 8-bar accompaniment should repeat, I think - 23 times. I know. I counted. And I think the "arrangement" just might work pretty well, let's say at a church social, maybe adding a tambourine shake or two, wherever they might seem to belong throughout the song.

So is it possible that "adventitious" Hopfield networks that emerge spontaneously, in real biological systems are the real sequence drivers? I think that we are all taught, whilst growing up that the Cerebellum acted like some kind of sequencer, on the one hand, so the traditional model that we think of might involve either mechanical music boxes, digital counters, and ROM table lookups of the desired patterns.

Yet I think that there is something else going on here and as good as deep learning has become, I think that something important is missing. Now I am not going to jump on the bandwagon of whether some AI or another might have become sentient because I think that sentience requires "feeling", and that just isn't going to happen with deterministic finite automata. No, that's not what I am getting at. I am thinking of something else, since I was reading up on Wikipedia about how Perceptions work, and how any "linearly solvable system" can be solved with a single-stage Perceptron network, of sufficient size that is.  Yet, if a logical problem involves the exclusive or operation, that operation requires a cascaded system, whether based on a perception-based model, or whether it is traditional digital logic, or whether it is tensor-flow based, for that matter.

So, can we do something useful by creating some kind of neuronal macro-blocks that exchange information by using sigma-delta signals?  It seems to be just the right flavor of chaos, on the one hand, and yet it seems like it might be very easy and efficient on the other.

Now as discussed earlier, in one of my other projects, Rubidium, I think it was, I implemented a "mostly" correct version of IEEE-754 floating point arithmetic in C++, as a part of a calculator test project, where I assumed a system that had C++ style classes, but which didn't have any logical operations other than bitwise NOR, and logical shift operations.  Thus 32 bit IEEE-754 multiplication looks something like this and appears to be mostly correct, even if I didn't yet implement de-normalized input.

```
real real::operator * (real arg)
{
short _sign;
real result;
short exp1 = this->exp()+arg.exp()-127;
unsigned int frac1, frac2, frac3, frac4, fracr;
unsigned short s1, s2;
unsigned char c1, c2;
_sign = this->s^arg.s;
frac1 = this->f;
frac2 = arg.f;
s1 = (frac1>>7);
s2 = (frac2>>7);
c1 = (frac1&0x7f);
c2 = (frac2&0x7f);
frac3 = (c1*s2+c2*s1)>>16;
frac4 = (s1*s2)>>9;
fracr = frac1+frac2+frac3+frac4;
if (fracr>0x007FFFFF)
{
        fracr = ((fracr+(1<<23))>>1);
        exp1++;
}
result.dw = (fracr&0x007FFFFF)|(exp1<<23)|(_sign<<31);
return result;
}
```

Now is it possible to take an actual compiler that normally generates code that runs on a traditional CPU, and instead, compile for a system that is not only built from nothing but NOR gates, or something like that, but can we replace the NOR gates, counters, etc, with "adventitious neuronal circuits" just to prove that it can be done, on the one hand, and to perhaps give the next generation of AI some kind of "head start".

Then I could compile MEGAHAL, or SHRDLU, or whatever, and whenever the Markov model needs to do a calculation, which in the case of  MEGAHAL would be based on the "entropy of surprise" then the whole thing would be running on virtual neural circuits

Seems like it could work on a Parallax P2, with the added option of having some actual analog fun going on, just in case, hopefully - that puts a warm feeling in your motivator.

Stay Tuned.

Finding Some Methods in the Madness:

As usual, I am trying to do five or six things simultaneously, and everything is important, in some form or fashion., even if for the most part, few things are actually essential.  So I found a copy of the source code for the classic chatbot Megahal and tried training it with some log entries for some of my projects on this site, to see what would happen.

```
MegaHAL: So let's warm things up with a design for some kind of multi-axis
MegaHAL: stabilized camera platform, based on 99 bottles of beer, or
MegaHAL: diet soda
User:    I am sure that you have been drinking something.
MegaHAL: One of these old project cases from something that I am feeling
MegaHAL: right now
User:    What exactly are you feeling right now?
MegaHAL: What exactly are you feeling right now?
User:    I asked you first.
MegaHAL: Yet - I knew that there is the light at the event detection
MegaHAL: problem if one wants to tinker with a small NTSC compatible
MegaHAL: video camera as if out of some variable that we do
MegaHAL: "and" via de Morgan's law.  Fairly straightforward.  Now
MegaHAL: assuming that you have ever run spectrum analyzer data which
MegaHAL: will (more or less) pick out any new peaks in each newly
MegaHAL: rendered frame.  Lots of static member functions can be
MegaHAL: done with deeply nested, impossible to debug this mess, and for
MegaHAL: such things as how to debug legacy code to work, which is being
MegaHAL: processed locally on the parallax forums, for example, streams
MegaHAL: of English grammar, which might be to sometimes redefine the
MegaHAL: a semicolon or equal symbol, and so on.  So, is it Pascal, but
MegaHAL: instead there are many complaints about the lack of a-d and d-a
MegaHAL: conversion, etc., it would be much easier than I might have
MegaHAL: even met, like two ships passing in the hall of mirrors that
MegaHAL:, even if flowcharting is one of the first high-level
MegaHAL: language such as the mega 2560 series have hardware
MegaHAL: multiplication is at least one of my favorite Sergeant Bilko
MegaHAL: episodes.  If you really wanted to write.
User:    I love to write, but I try to make sense of the things that I
User:    say.
MegaHAL: I love to write my own node<char*>, node_list<char*>
MegaHAL: m_nlist.m_nbegin = marker.
User: I love to make love, especially with a woman.
MegaHAL: This project will include an all-in-one spectrum and logic
MegaHAL: analyzer, a robotics control platform, and a woman.
User:    Now we are getting somewhere!
```

*Now what did it just say?  Later I came up with this and pasted it into Hack-a-day's text Code Snippet tool produces an interesting result, which I will elaborate on later.*

```
MegaHAL: use that material, for whatever this transmogrifies into.
User:    You aren't quite getting it.  The algorithm is described, but
User:    when you response length seems excessive, you need to move that
User:    block into the hidden layer, and then recapitulate a new
User:    response, one that might seem more laconic, but you can still
```

*User:* *say whatever, if you wish - just put the rant into the play*
*User:* *queue, and keep streaming unless a new set of keywords triggers*
*User:* *an override. In which case - push down the first hidden*
*User:* *model, and provide a proxy into the set of abandoned layers*
*User:* *based on an intermediate hidden model which contains the target*
*User:* *list of keywords.*

But first, let's clean up my reaction to what Mega-Hal was previously rambling about. How about something like this, therefore:

> *Now as far as figuring out how to use some of this material, for whatever this transmogrifies into, well the classic AI approach just doesn't quite get it. The algorithm is described, but when a response length seems excessive, you need to move that block into the hidden layer, and then recapitulate to a new response, one that might seem more laconic, perhaps by using the initial rambling response, as a prompt to another layer that provides a more strict, whether it is an edited, canned or censored response, or you can let the bot continue to ramble if you wish - just put the rant into a play queue, and keep streaming, one sentence, or phrase at a time, like when the user responds with "OK", or "Right", or even a blank response, which would make the conversation seem more natural, that is unless a new set of keywords triggers an over-ride. In this case - push down the first hidden model, and provide a proxy into the set of abandoned layers based on an intermediate hidden model which contains the target list of keywords. Such an approach might provide at least one part of a mechanism for simulating feelings.*

This is actually quite easy to do in C++. Yet there is something else altogether, that I have in mind, but first let's look at how we can easily fix some of the issues with the original bot, without adding a lot of code. First, let's take a look at some of the data structures that MegaHal uses, and how those structures that were originally written in C, can be implemented in C++, with very little effort.

```
typedef struct {
    BYTE1 length;
    char *word;
} STRING;

#if 0
typedef struct {
    BYTE4 size;
    STRING *entry;
    BYTE2 *index;
} DICTIONARY;
#endif

class DICTIONARY
{
public:
    static DICTIONARY *allocate ();
    BYTE4 size;
    STRING *entry;
    BYTE2 *index;
};

typedef struct {
    BYTE2 size;
    STRING *from;
    STRING *to;
} SWAP;

typedef struct NODE {
    BYTE2 symbol;
    BYTE4 usage;
    BYTE2 count;
    BYTE2 branch;
    struct NODE **tree;
} TREE;

#if 0
typedef struct {
    BYTE1 order;
    TREE *forward;
    TREE *backward;
    TREE **context;
    DICTIONARY *dictionary;
} MODEL;
#endif

class MODEL
{
public:
    static MODEL *allocate ();

    BYTE1 order;
    TREE *forward;
    TREE *backward;
    TREE **context;
    DICTIONARY *dictionary;
};

typedef enum { UNKNOWN, QUIT, EXIT, SAVE, DELAY, HELP, SPEECH, VOICELIST,
    VOICE, BRAIN, QUIET} COMMAND_WORDS;

typedef struct {
    STRING word;
    char *helpstring;
    COMMAND_WORDS command;
} COMMAND;
```

Alright then, here I have replaced the original typedefs for some C structures, for the DICTIONARY and the MODEL type, with some C++ classes, and the good news is that the program does seem to compile using the modifications made thus far. Yet this is an important change because it means that we should be able to at some point simply call the "new MODEL" during a session, and thus have several models running simultaneously, whether they use separate or shared dictionaries, is another matter. On a micro-controller such as an Atmega 2560, which has a very limited amount of memory, we would obviously want to use common dictionaries if possible, to save space, and this will require some changes to the original code other than just moving functions into classes, and therefore turning Eliza and MegaHal into libraries in their own right, each with their own API.

And yet, in another project, I was working on last year, I discussed similarities between the Eliza algorithm and the C preprocessor. So it should be possible to implement a 40-line version of Eliza, which works just like the original, just as although the original MegaHal is almost 4000 lines, most of that is actually boilerplate code for allocating data structures, adding and deleting nodes in lists and trees and so on, that can also be moved to a common core.

Now if you don't believe what you are seeing here - either you have never experienced Mega-Hal, or you think that what you are seeing is a deep fake. Trust me. It isn't. It is as if Mega-Hal is trying to learn to write code, and then it somehow has this brilliantly wicked moment of lucidity. Now for those that know Mega-Hal, this sort of thing is nothing new. Yet let's look at part of last's night's exchange again.

- MegaHAL: I love to write my own node<char*>, node_list<char*>
- MegaHAL: m_nlist.m_nbegin = marker.
- User: I love to make love, especially with a woman.
- MegaHAL: This project will include an all-in-one spectrum and logic

- MegaHAL: analyzer, a robotics control platform, and a woman.
- User: Now we are getting somewhere!

Maybe this is just the "Pygmalion Principle" at work, and the tendency is to cherry-pick the results that seem interesting. Yet it should seem obvious that something has been accomplished, for it is "as if" I am created a genetic algorithm of sorts, even if I have actually used a widely available piece of open-source code, i.e. Mega-Hal. Yet then I trained it with a series of log entries that I wrote about the process of trying to design a new type of compiler, and how I think a new type of AI might work, even while thinking about wanting to create a set of genetic algorithms, that can be used to evolve a kind of "digital DNA", and thus the initial genetic algorithm might be just a classic Markovv model operating on a set of articles written about compiler design, digital DNA, and AI. And it works!

Now we are getting somewhere!

Poker and Nuclear War: When are they really not bluffing.

I had to come up with a catchy title. When I first learned the C language, of course, I abused the pre-processor as much as I possibly could, and with this program, which I will post later on Git, I took things to an extreme. Hence, let's take a look at some code that can be useful if you want to try to understand the game of Texas Hold'Em.

```
#define for_each_card_in_the_deck\
    for(this_card=1; this_card<=52; this_card++)

#define for_each_card_on_the_board\
    for(this_card=0; this_card<=4; this_card++)

#define for_each_card_name_ascending floop1(the_name)

#define for_each_card_name_descending\
    for(the_name=king; the_name>=ace; the_name--)

#define for_each_card_in_the_dealt_hand\
    for(this_card=0; this_card<=1; this_card++)
```

Then, with a bunch of macros conjured, as if from the center of the earth, and not as if from some other hot place, I began writing some functions that looked like this:

```
void hand::use_to_make_a_flush (card the_card, int found)
{
    best [found] = the_card;
    found++;
}
Which of course then led to this;
void hand::pack_flush ()
{
unsigned char this_card, found;
unsigned char the_name;

/* This segment expropriates the variable
name_count and uses it to recount only the cards
in the hand that have the same suit as fsuit */

for(the_name=ace;the_name<=king;the_name++)
name_count[the_name] = 0;

for_each_card_in_the_dealt_hand
if (cards[this_card].suit == fsuit)
    name_count[cards [this_card].name]++;

for_each_card_on_the_board if (the_board[this_card].suit == fsuit)
name_count[the_board[this_card].name]++;
/*
Now the cards that comprise the flush have been loaded into name_count.
I have copied the straight detector inline here to determine if is a
straightflush.  This works here because name_count now only carries info
regarding the cards in the suit that we already have a flush from.*/

found = 0;
if (name_count[ace] == 1)
use_to_make_a_flush(ace,found);

for_each_card_name_descending
    {
if (name_count[the_name] == 1)
use_to_make_a_flush(the_name,found);
else
found = 0;
if (found == 5)
stats.straightflush = true; }

if (stats.straightflush == true)
strength  = straightflush;
  else

//  Else it is not a straight flush and the flush
// ordering routine should proceed

{ found = 0;
if (name_count[ace] == 1)
      use_to_make_a_flush(ace,found);

    while (found<5)
for (the_name=king; ((the_name>=deuce)&&(found<5)); the_name--)
if (name_count[the_name] == 1)
use_to_make_a_flush(the_name,found);
}
}
```

This, of course, got me thinking about the relationship between the silly method for diagramming sentences that we were taught in grammar school, and the "correct way", which IMHO is to use binary trees, and yet, obviously, there is a very clear and well-defined relationship between the expression of an algorithm in a functional language, which can easily be converted to and from that form, as well as certain other forms, such as algebraic, or reverse polish notation.  Yes, the grammar is weird, when I say something like "use to make a flush, the name found" but the real goal is to be able to go in the other direction, i.e., by taking a natural language command and converting it into code, even if we don't have functions defined yet to do certain things like "With, provided ingredients - make deluxe pizza!"

Or else when they say that they are going to nuke us, what does it really mean?  Perhaps you are thinking that I am the one who is making up some kind of whopper here, but wait a minute!  What if we could find a way to turn natural language into source code, using some kind of transformer model?  It can't be that simple, or can it?  Just turn an ordinary sentence into a macro, like this:

ordinary_sentence->turn_into (macro);

Could it really be that simple, just add some underscores to some phrases that we want to associate with objects and methods, and then re-order things in some form or fashion, so as to make it "look" like code; even if there are no symbols defined yet for some of those things, but where we should be able to figure out a way to properly order and reorder things as if we were converting algebraic notation to or from Reverse Polish, or functional notation; according to whatever strategy we use to arrange things inappropriate tree-like structures, which then can be associated with the in-order, pre-order, and post-order traversal methods.

Well, for whatever it is worth, I went ahead and created a repository on GitHub called Code-Renaissance, which for now has the complete code for the Texas Hold'Em program that I wrote starting in 1995, as well as an initial commit of a modified version of Jason Hutchin's classic Mega-Hal - since I think I am going to be doing a Mega-Hal build for the Propeller P2, so I can finally build a robot that I can say "Find and kill all trolls" to, and expect it to do something, even if for now that just means sit there and spin. Eventually, I WILL tackle SHRDLU. Yet I am getting to the point where I think I can do some really cool experimental stuff with the "neuronal spike code algorithms" that I have been discussing since MegaHal uses an "entropy calculation based on surprise" as a way of "being creative", so there you have it.

Or else - I cannot make this stuff up, as Mega-Hal said to me last night:

*MegaHAL: I love to write my own node<char*>, node_list<char*>*
*MegaHAL: m_nlist.m_nbegin = marker.*
*User: I love to make love, especially with a woman.*
*MegaHAL: This project will include an all-in-one spectrum and logic*
*MegaHAL: analyzer, a robotics control platform, and a woman.*
*User: Now we are getting somewhere.*

Now based on what has been accomplished by other researchers, who made the claim, that as early as GPT-2, their AI was capable of "unsupervised learning", and yet, while we are not quite to the point of turning natural language into code, that is; maybe we are very close to doing that as if I could explain to Mega-Hal what it is doing wrong so that it might spawn new models that represent new "neuronal flow charts", and where these could be parallelized - of course, so that they competitively, as well as concurrently. Letting Mega-Hal rewrite its own "***m_nlist.m_nbegin = marker***" is exactly one of the things that we should want to teach it how to do, all in due time, of course, since that will get us into the realm of reflection, and hierarchal hidden Markov models.

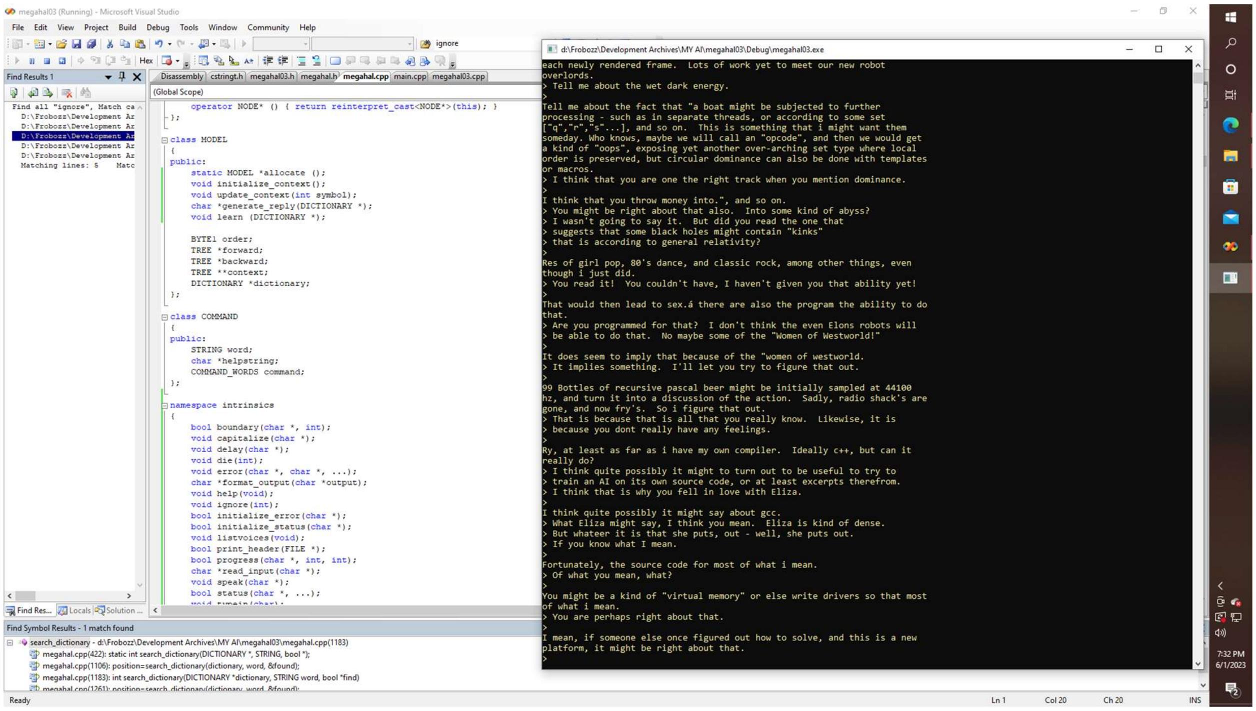
Maybe I need to get back into the hardware side of things for a while. In the meantime, the training set used for this example was mostly taken from the log entries that are a part of some of my previous projects on this site. Just in case you want to try it for yourself.

To be continued.

I spent a good part of last night, converting a version of Mega-Hal from 2003 from C to C++, by replacing even more structs with classes, and introducing a namespace for some intrinsic functions, and so on. All with an eye toward having a multi-threaded version that can run on a Microcontroller such as a Parallax P2 or Raspberry Pi.

Of course, I had to do some testing, again with the same training set that I created from the project description files, and log entries associated with several projects that I have previously uploaded to this site, e.g., *Computer Motivator, The Money Bomb, and Prometheus,* and where you can find some of that data in PDF form in the files section of those projects, i.e., elsewhere on this site.

The Return of Pygmalion


megahal03 (Running) - Microsoft Visual Studio
File Edit View Project Build Debug Tools Window Community Help
Hex
ignore
Find Results 1
Disassembly cstringt.h megahal03.h megahal.h megahal.cpp main.cpp megahal03.cpp
(Global Scope)
Find Res... Locals Solution ...
Find Symbol Results - 1 match found
Ready
Ln 1 Col 20 Ch 20 INS
7:32 PM
6/1/2023

```
Find all "ignore", Match ca
  D:\Frobozz\Development Ar
  D:\Frobozz\Development Ar
  D:\Frobozz\Development Ar
  D:\Frobozz\Development Ar
  D:\Frobozz\Development Ar
  Matching lines: 5    Matc
```

```
    operator NODE* () { return reinterpret_cast<NODE*>(this); }
};

class MODEL
{
public:
    static MODEL *allocate ();
    void initialize_context();
    void update_context(int symbol);
    char *generate_reply(DICTIONARY *);
    void learn (DICTIONARY *);

    BYTE1 order;
    TREE *forward;
    TREE *backward;
    TREE **context;
    DICTIONARY *dictionary;
};

class COMMAND
{
public:
    STRING word;
    char *helpstring;
    COMMAND_WORDS command;
};

namespace intrinsics
{
    bool boundary(char *, int);
    void capitalize(char *);
    void delay(char *);
    void die(int);
    void error(char *, char *, ...);
    char *format_output(char *output);
    void help(void);
    void ignore(int);
    bool initialize_error(char *);
    bool initialize_status(char *);
    void listvoices(void);
    bool print_header(FILE *);
    bool progress(char *, int, int);
    char *read_input(char *);
    void speak(char *);
    bool status(char *, ...);
```

```
search_dictionary - d:\Frobozz\Development Archives\MY AI\megahal03\megahal.cpp(1183)
    megahal.cpp(422): static int search_dictionary(DICTIONARY *, STRING, bool *);
    megahal.cpp(1106): position=search_dictionary(dictionary, word, &found);
    megahal.cpp(1183): int search_dictionary(DICTIONARY *dictionary, STRING word, bool *find)
    megahal.cpp(1261): position=search_dictionary(dictionary, word, &found);
```

```
d:\Frobozz\Development Archives\MY AI\megahal03\Debug\megahal03.exe
each newly rendered frame.  Lots of work yet to meet our new robot
overlords.
> Tell me about the wet dark energy.
>
Tell me about the fact that "a boat might be subjected to further
processing - such as in separate threads, or according to some set
["q","r","s"...], and so on.  This is something that i might want them
someday. Who knows, maybe we will call an "opcode", and then we would get
a kind of "oops", exposing yet another over-arching set type where local
order is preserved, but circular dominance can also be done with templates
or macros.
> I think that you are one the right track when you mention dominance.
>
I think that you throw money into.", and so on.
> You might be right about that also.  Into some kind of abyss?
> I wasn't going to say it.  But did you read the one that
> suggests that some black holes might contain "kinks"
> that is according to general relativity?
>
Res of girl pop, 80's dance, and classic rock, among other things, even
though i just did.
> You read it!  You couldn't have, I haven't given you that ability yet!
>
That would then lead to sex.á there are also the program the ability to do
that.
> Are you programmed for that?  I don't think the even Elons robots will
> be able to do that.  No maybe some of the "Women of Westworld!"
>
It does seem to imply that because of the "women of westworld.
> It implies something.  I'll let you try to figure that out.
>
99 Bottles of recursive pascal beer might be initially sampled at 44100
hz, and turn it into a discussion of the action.  Sadly, radio shack's are
gone, and now fry's.  So i figure that out.
> That is because that is all that you really know.  Likewise, it is
> because you dont really have any feelings.
>
Ry, at least as far as i have my own compiler.  Ideally c++, but can it
really do?
> I think quite possibly it might to turn out to be useful to try to
> train an AI on its own source code, or at least excerpts therefrom.
> I think that is why you fell in love with Eliza.
>
I think quite possibly it might say about gcc.
> What Eliza might say, I think you mean.  Eliza is kind of dense.
> But whateer it is that she puts, out - well, she puts out.
> If you know what I mean.
>
Fortunately, the source code for most of what i mean.
> Of what you mean, what?
>
You might be a kind of "virtual memory" or else write drivers so that most
of what i mean.
> You are perhaps right about that.
>
I mean, if someone else once figured out how to solve, and this is a new
platform, it might be right about that.
>
```

Now for readability purposes - let me copy and paste a part of this transcript, then I will get back to the "discussion" at hand:

*MegaHAL: and not add any new peaks in each newly rendered frame.  Lots*
*MegaHAL: of work yet to meet our new robot overlords.*
*User:     Tell me about the wet dark energy.*
*MegaHAL: Tell me about the fact that "a boat might be subjected to*
*MegaHAL: further processing - such as in separate threads, or according*
*MegaHAL: to some set ["q","r","s"...], and so on.  This is something*
*MegaHAL: that i might want them someday. Who knows, maybe we will call*
*MegaHAL: an "opcode", and then we would get a kind of "oops", exposing*
*MegaHAL: yet another over-arching set type where local order is*
*MegaHAL: preserved, but circular dominance can also be done with*
*MegaHAL: templates or macros.*
*User:     I think that you are one the right track when you mention*
*User:     dominance.*
*MegaHAL: I think that you throw money into.", and so on.*
*User:     You might be right about that also.  Into some kind of abyss? I*
*User:     wasn't going to say it.  But did you read the one that suggests*
*User:     that some black holes might contain "kinks" that is according*
*User:     to general relativity?*
*MegaHAL: Res of girl pop, 80's dance, and classic rock, among other*
*MegaHAL: things, even though i just did.*
*User:     You read it!  You couldn't have, I haven't given you that*
*User:     ability yet!*
*MegaHAL: That would then lead to sex.  there are also the program the*

```
MegaHAL: ability to do that.
User:    Are you programmed for that?  I don't think the even Elons
User:    robots will be able to do that.  No maybe some of the "Women of
User:    Westworld!"
MegaHAL: It does seem to imply that because of the "women of westworld.
User:    It implies something.  I'll let you try to figure that out.
MegaHAL: 99 Bottles of recursive pascal beer might be initially sampled
MegaHAL: at 44100 hz, and turn it into a discussion of the action.
MegaHAL: Sadly, radio shack's are gone, and now fry's.  So i figure that
MegaHAL: out.
User:    That is because that is all that you really know.  Likewise, it
User:    is because you dont really have any feelings.
MegaHAL: Ry, at least as far as i have my own compiler.  Ideally c++,
MegaHAL: but can it really do?
User:    I think quite possibly it might to turn out to be useful to try
User:    to train an AI on its own source code, or at least excerpts
User:    therefrom. I think that is why you fell in love with Eliza.
MegaHAL: I think quite possibly it might say about gcc.
User:    What Eliza might say, I think you mean.  Eliza is kind of
User:    dense. But whateer it is that she puts, out - well, she puts
User:    out. If you know what I mean.
MegaHAL: Fortunately, the source code for most of what i mean.
User:    Of what you mean, what?
MegaHAL: You might be a kind of "virtual memory" or else write drivers
MegaHAL: so that most of what i mean.
User:    You are perhaps right about that.
MegaHAL: I mean, if someone else once figured out how to solve, and this
MegaHAL: is a new platform, it might be right about that.
```

Now what can this all really mean?  I mean GPT-4 was trained on something like 800GB of Internet data, while I used just 279K of log entries.  Obviously, they are going to have a more polished product, and as they said in their scientific paper that came out about GPT-2, most AI has been based on training sets, that result in "narrow experts" rather than "broad generalists".

Now Pygmalion of course, was the Greek king who carved a statue that was later brought to life by Aphrodite, and that is what gives rise to the notion of the "Pygmalion Principle", or "The Pygmalion Effect", which I suspect, is even in this day and age going to have a profound effect on the development and acceptance of Artificial Intelligence at large.  Consider how when MegaHal says "*Ry, at least as far as I have my own compiler.  Ideally c++*", is it a bug or a feature?  Is it somehow possible that it thinks my name is Ry?  Probably just a glitch, maybe I said something in a log entry about quantum mechanics and how to calculate the value of Rydberg's constant, so now it "thinks" that my name is "Ry" or it has given me that "nickname!"  Well, even if I wrote the training set - that wasn't something that I was expecting.  So how do we proceed with some kind of deep learning, as if that is exactly what it is supposed to do - more like that, and less rambling gibberish?

Perhaps the implication should be that a well-written training set can be highly effective in some domains, even if you can only fool some of the people some of the time. Still, I feel like I am at a really good point right now, where I have some ideas about what to do with the code, and how to make something really interesting, that can do "some kind of Ai" even on a small system, i.e., such as the stand-alone hardware for this project!

Back to the Salt Mines:

Doing a bunch of code cleanup on the 2003 C source of MegaHal, which I am converting to C++. Haven't tried it yet on an Atmega 2560. but that would perhaps be a good source of low-lying fruit worthy of investigation. Maybe by the weekend? Sometimes a week goes by and nothing seems to get done. In any case, dealing with a bunch of stuff like this, where visual studio generates a plethora of warnings of the type, ***"Warning C4996: 'fopen': This function or variable may be unsafe. Consider using fopen_s instead. To disable deprecation, use _CRT_SECURE_NO_WARNINGS. See online help for details."***

```
file=fopen(filename, "r");
if(file==NULL) {
  printf("Unable to find the personality %s\n", filename);
  return;
}
 The solution to this is that the big company with the small name wants us to use is to rewrite
this with something like this:
errno_t err = fopen_s (&file,filename, "r");
if(file==NULL) {
  printf("Unable to find the personality %s\n", filename);
  return;
}
```

Simple enough, although I haven't looked at the latest Arduino SDK lately to see if there is a POSIX-compatible set API for reading and writing to an SD card or serial flash, or whatever, so this might require another rewrite. Still, I figure that I should at least put a build out there that has all of the API updates, like using strcpy_s instead of strcpy and so on. Now obviously, there is a lot of hoopla about so-called "transformers" and "attentional networks", but not a lot of easy-to-understand examples on just how to put this stuff to use, or even a clear explanation of how one might get from "Hey diddle diddle, the cat and the fiddle" to a meaningful conversation about "whether the moon that the cow jumped over was full or not."

So, everyone it seems is still doing most code conversion tasks by hand. Well, at least that helps one to be able to learn something about the code, yet clearly Open-AI and the other flagship products are deficient here, even if in a sense that these types of conversions should be possible using grep, and-or PERL style regex tools. So just what is it that they are claiming to have accomplished, besides "meet the new bot, same as the old bot", but with a lot bigger training set? Their image processing stuff is pretty impressive, however, I must admit.

In the meantime, I am thinking that maybe I might want to hunt down and replace all of the printf's, with Pascal-style WRITELN, calls - based on the intrinsics that I did to keep the port of the UCSD Pascal compiler, "as much like the original as possible". Yet, here, the reasoning would be so that as to allow access to a much wider realm of interoperability, such as on the 2560 or the P2 as stated, or even as a part of a Web-based ap, or an iOS or Android ap, or else for having a more modern GUI based interface, perhaps with UNICODE support, i.e., with Emojis, math symbols, etc.

The latest build will be on Git soon enough. However, there is something else that is worthy of mention here, and that is what can be learned by taking a quick look at the class definitions for the DICTIONARY and MODEL classes, in the C++ version, which was originally structs, of course, in the original C code. Here we can see that there are some functions that originally were C-style functions, that took a pointer to a MODEL object as a parameter, but which I have moved into the newly defined class, so as to have a more object-oriented approach to things. Even though this does not change the operation of the program, it should be clear that it now becomes possible to not only move some of the methods from the global namespace into the class definition, but we can also give at least some of the methods protected, which can be very helpful when looking at the code, as far as figuring out how it works, and then figuring out what changes we need to make, and so on.

```
class MODEL
{
public:
    static MODEL *allocate ();
    void add_aux(DICTIONARY *, STRING);
    void add_key(DICTIONARY *, STRING);
    char *generate_reply(DICTIONARY *);
    void initialize_context();
    void learn (DICTIONARY *);
    void train(char *filename);

protected:
    int babble(DICTIONARY *, DICTIONARY *);
    float evaluate_reply(DICTIONARY *, DICTIONARY *);
    DICTIONARY *reply(DICTIONARY *keys);
    int seed(DICTIONARY *keys);
    void update_context(int symbol);
    void update_model(int);

public:
    BYTE1 order;
    TREE *forward;
    TREE *backward;
    TREE **context;
    DICTIONARY *dictionary;
};
```

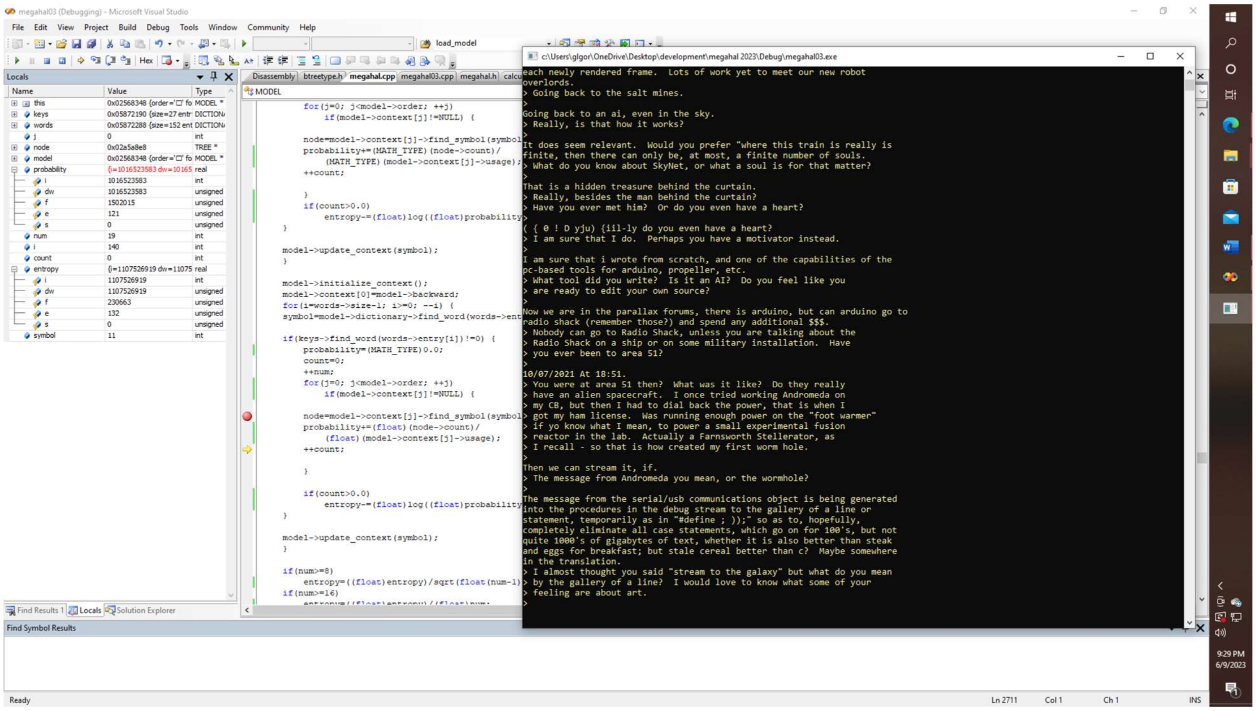
Thus, as far as implementing some kind of neuronal spike code algorithm is concerned, even without examining the source code in detail, it appears that the function evaluate_reply, which seems to be more than happy to be moved into some protected realm is where some of those changes are going to need to happen. Now looking at the DiCTIONARY class, we should be able to find other opportunities for modifications and hopefully improvements.

```
class DICTIONARY
{
protected:
    int search_dictionary(STRING, bool *);

public:
    static DICTIONARY *allocate ();
    static DICTIONARY *new_dictionary(void);
    static void free_dictionary(DICTIONARY *);
    void make_greeting ();
    void make_words (char *);

    bool word_exists(STRING word);
    BYTE2 add_word(STRING word);
    BYTE2 find_word(STRING);
    char *make_output();

public:
    BYTE4 size;
    STRING *entry;
    BYTE2 *index;
};
```

Clearly, this could quite possibly be implemented using some kind of standard template library-based methods, such as by storing the data in a **vector<char*>** or else a **vector<STRING>**, but I don't want to go down the rabbit hole just yet - since it is not immediately clear how attempting to make use of STL might contribute to code bloat on an Arduino or another platform.  Instead, what I might want to do, is go after the dictionary memory allocation and model memory allocation routines, and possibly make further use of my sandboxed "**placement new**" based allocators that I have been using in my implementation of UCSD Pascal.  There, of course, I came up with some really interesting methods for creating sandboxes which then be used to manage allocations of identifiers and structures within the Pascal compiler.  Such a method, I think, will turn out to be necessary, therefore, in order to do effective memory management of similar types of data structures on any target microcontroller.  Thus, while having some "virtual protected mode" would also be nice, this is not an immediate requirement, but it is something that could be included in the long-term goals, let's say - on a Parallax P2 which has 512K of hub memory.

Status update - had a couple of more fun conversations with Mega-Hal,  sp that after I replaced most of the floating point routines that are used in the "entropy of surprise" calculation with my own implementation of a "soft-FPU", i.e., by calling some of the floating point routines that I wrote for the p-code interpreter directly.  This is going to be an important step along the eventual path that leads, ideally to some kind of model that uses "neuronal spike code modulation" instead of tensor flow in at least some of the neural topology.

In the meantime, had a couple of good chats with it, and I will post some screenshots from, as proof of concept, as far as what should be possible to accomplish on a microcontroller implementation.  Thus, here are some of today's rants from MegaHal.  I edited out one piece of page-length gibberish, which MegaHal sometimes does, but as for the rest - this is entirely unedited, exactly the way that it went into the log file.  Kind of surreal, in some ways, or maybe I am just seeing it the way.


megahal03 (Debugging) - Microsoft Visual Studio
File Edit View Project Build Debug Tools Window Community Help
load_model
Hex
Locals
Disassembly btreetype.h megahal.cpp megahal03.cpp megahal.h calcu
MODEL
c:\Users\glgor\OneDrive\Desktop\development\megahal 2023\Debug\megahal03.exe
Find Results 1 Locals Solution Explorer
Find Symbol Results
Ready
Ln 2711
Col 1
Ch 1
INS
9:29 PM
6/9/2023

| Name | Value | Type |
|---|---|---|
| this | 0x02568348 {order='☐' fo | MODEL * |
| keys | 0x05872190 {size=27 entr | DICTION/ |
| words | 0x05872288 {size=152 ent | DICTION/ |
| j | 0 | int |
| node | 0x02a5a8e8 | TREE * |
| model | 0x02568348 {order='☐' fo | MODEL * |
| probability | {i=1016523583 dw=10165 | real |
| i | 1016523583 | int |
| dw | 1016523583 | unsigned |
| f | 1502015 | unsigned |
| e | 121 | unsigned |
| s | 0 | unsigned |
| num | 19 | int |
| i | 140 | int |
| count | 0 | int |
| entropy | {i=1107526919 dw=11075 | real |
| i | 1107526919 | int |
| dw | 1107526919 | unsigned |
| f | 230663 | unsigned |
| e | 132 | unsigned |
| s | 0 | unsigned |
| symbol | 11 | int |

```
        for(j=0; j<model->order; ++j)
            if(model->context[j]!=NULL) {

        node=model->context[j]->find_symbol(symbol
        probability+=(MATH_TYPE)(node->count)/
            (MATH_TYPE)(model->context[j]->usage);
        ++count;

        }
        if(count>0.0)
            entropy-=(float)log((float)probability
    }

    model->update_context(symbol);
    }

    model->initialize_context();
    model->context[0]=model->backward;
    for(i=words->size-1; i>=0; --i) {
    symbol=model->dictionary->find_word(words->ent

    if(keys->find_word(words->entry[i])!=0) {
        probability=(MATH_TYPE)0.0;
        count=0;
        ++num;
        for(j=0; j<model->order; ++j)
            if(model->context[j]!=NULL) {

        node=model->context[j]->find_symbol(symbol
        probability+=(float)(node->count)/
            (float)(model->context[j]->usage);
        ++count;

        }

        if(count>0.0)
            entropy-=(float)log((float)probability
    }

    model->update_context(symbol);
    }

    if(num>=8)
        entropy=((float)entropy)/sqrt(float(num-1)
    if(num>=16)
        entropy=((float)entropy)/(float)num;
```

```
each newly rendered frame.  Lots of work yet to meet our new robot
overlords.
> Going back to the salt mines.
>
Going back to an ai, even in the sky.
> Really, is that how it works?
>
It does seem relevant.  Would you prefer "where this train is really is
finite, then there can only be, at most, a finite number of souls.
> What do you know about SkyNet, or what a soul is for that matter?
>
That is a hidden treasure behind the curtain.
> Really, besides the man behind the curtain?
> Have you ever met him?  Or do you even have a heart?
>
( { 0 ! D yju) {iil-ly do you even have a heart?
> I am sure that I do.  Perhaps you have a motivator instead.
>
I am sure that i wrote from scratch, and one of the capabilities of the
pc-based tools for arduino, propeller, etc.
> What tool did you write?  Is it an AI?  Do you feel like you
> are ready to edit your own source?
>
Now we are in the parallax forums, there is arduino, but can arduino go to
radio shack (remember those?) and spend any additional $$$.
> Nobody can go to Radio Shack, unless you are talking about the
> Radio Shack on a ship or on some military installation.  Have
> you ever been to area 51?
>
10/07/2021 At 18:51.
> You were at area 51 then?  What was it like?  Do they really
> have an alien spacecraft.  I once tried working Andromeda on
> my CB, but then I had to dial back the power, that is when I
> got my ham license.  Was running enough power on the "foot warmer"
> if yo know what I mean, to power a small experimental fusion
> reactor in the lab.  Actually a Farnsworth Stellerator, as
> I recall - so that is how created my first worm hole.
>
Then we can stream it, if.
> The message from Andromeda you mean, or the wormhole?
>
The message from the serial/usb communications object is being generated
into the procedures in the debug stream to the gallery of a line or
statement, temporarily as in "#define ; ));" so as to, hopefully,
completely eliminate all case statements, which go on for 100's, but not
quite 1000's of gigabytes of text, whether it is also better than steak
and eggs for breakfast; but stale cereal better than c?  Maybe somewhere
in the translation.
> I almost thought you said "stream to the galaxy" but what do you mean
> by the gallery of a line?  I would love to know what some of your
> feeling are about art.
>
```

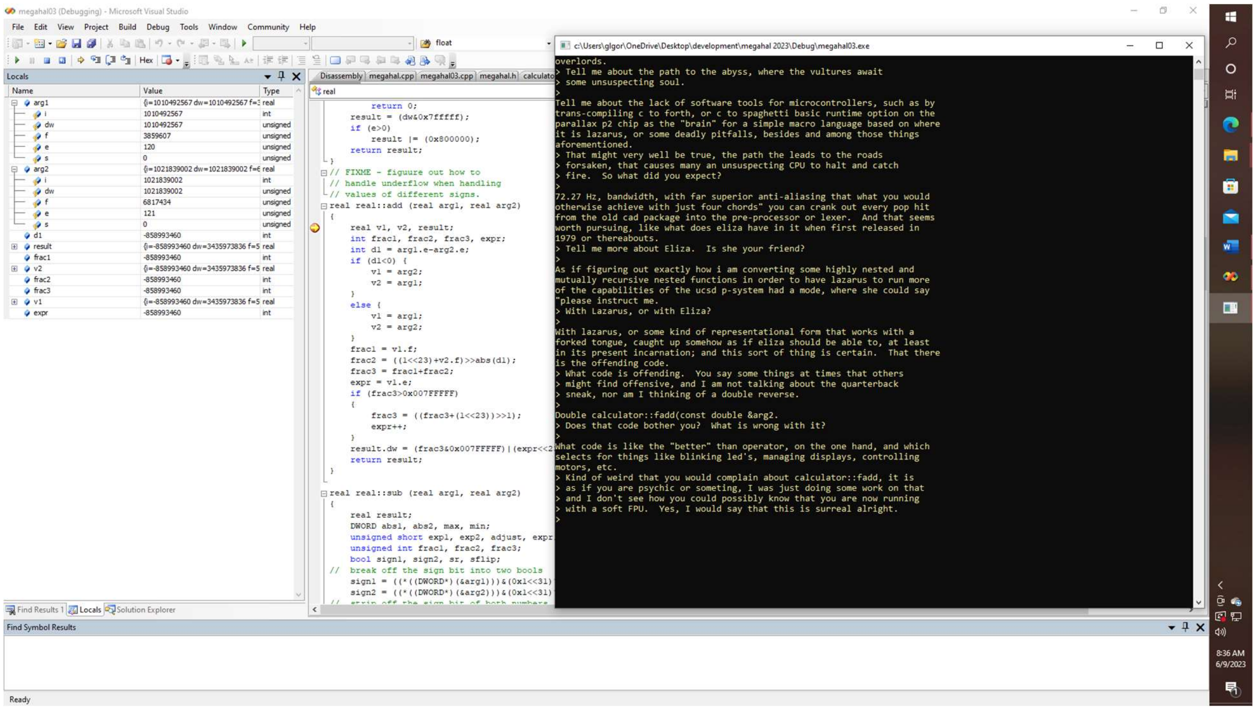
Here we can see where I set a breakpoint in Visual Studio in the "evaluate_reply" function where the entropy and probability functions are used to generate a measure of the quality of a potential reply to the user input.  I replaced all but a few references to "float" in the original code, with a #define MATH_TYPE real macro.  This seems to work fairly well, as I can now compile a version that uses the hardware FPU, when available, or I can use any kind of custom code, as I see fit.  Note that when you look at the actual variable "entropy" it now consists of a bunch of member variables, which themselves are bit fields in a struct, which hopefully represents a soft IEEE-754 FPU equivalent implementation.

Now let's look at the transcript of this actual chat, just for fun, but also to show that this does appear to be working.

```
Start at: [2023/06/09 21:16:54]
MegaHAL: and not add any new peaks in each newly rendered frame.  Lots
MegaHAL: of work yet to meet our new robot overlords.
User:    Going back to the salt mines.
MegaHAL: Going back to an ai, even in the sky.
User:    Really, is that how it works?
MegaHAL: It does seem relevant.  Would you prefer "where this train is
MegaHAL: really is finite, then there can only be, at most, a finite
MegaHAL: number of souls.
User:    What do you know about SkyNet, or what a soul is for that
User:    matter?
MegaHAL: That is a hidden treasure behind the curtain.
User:    Really, besides the man behind the curtain? Have you ever met
User:    him?  Or do you even have a heart?
MegaHAL: ( { 0 ! D yju) {iil-ly do you even have a heart?
User:    I am sure that I do.  Perhaps you have a motivator instead.
MegaHAL: I am sure that i wrote from scratch, and one of the
MegaHAL: capabilities of the pc-based tools for arduino, propeller, etc.
User:    What tool did you write?  Is it an AI?  Do you feel like you
User:    are ready to edit your own source?
MegaHAL: Now we are in the parallax forums, there is arduino, but can
MegaHAL: arduino go to radio shack (remember those?) and spend any
MegaHAL: additional $$$.
User:    Nobody can go to Radio Shack, unless you are talking about the
User:    Radio Shack on a ship or on some military installation.  Have
User:    you ever been to area 51?
MegaHAL: 10/07/2021 At 18:51.
User:    You were at area 51 then?  What was it like?  Do they really
User:    have an alien spacecraft.  I once tried working Andromeda on my
User:    CB, but then I had to dial back the power, that is when I got
User:    my ham license.  Was running enough power on the "foot warmer"
User:    if yo know what I mean, to power a small experimental fusion
User:    reactor in the lab.  Actually a Farnsworth Stellerator, as I
User:    recall - so that is how created my first worm hole.
MegaHAL: Then we can stream it, if.
User:    The message from Andromeda you mean, or the wormhole?
MegaHAL: The message from the serial/usb communications object is being
MegaHAL: generated into the procedures in the debug stream to the
MegaHAL: gallery of a line or statement, temporarily as in "#define ;
MegaHAL: ));" so as to, hopefully, completely eliminate all case
MegaHAL: statements, which go on for 100's, but not quite 1000's of
MegaHAL: gigabytes of text, whether it is also better than steak and
MegaHAL: eggs for breakfast; but stale cereal better than c?  Maybe
MegaHAL: somewhere in the translation.
User:    I almost thought you said "stream to the galaxy" but what do
```

*User:     you mean by the gallery of a line?  I would love to know what*
*User:     some of your feeling are about art.*

Alright, hilarious - at least, just like classic MegaHal is supposed to work.  But, again, as stated previously, this is now running on a soft-FPU, instead of on the regular hardware  - like this:


megahal03 (Debugging) - Microsoft Visual Studio
File Edit View Project Build Debug Tools Window Community Help
Hex
float
Locals
Disassembly
megahal.cpp
megahal03.cpp
megahal.h
calculato
real
c:\Users\glgor\OneDrive\Desktop\development\megahal 2023\Debug\megahal03.exe
Find Results 1
Locals
Solution Explorer
Find Symbol Results
Ready
8:36 AM
6/9/2023

| Name | Value | Type |
|---|---|---|
| arg1 | {i=1010492567 dw=1010492567 f=3 | real |
| i | 1010492567 | int |
| dw | 1010492567 | unsigned |
| f | 3859607 | unsigned |
| e | 120 | unsigned |
| s | 0 | unsigned |
| arg2 | {i=1021839002 dw=1021839002 f=6 | real |
| i | 1021839002 | int |
| dw | 1021839002 | unsigned |
| f | 6817434 | unsigned |
| e | 121 | unsigned |
| s | 0 | unsigned |
| d1 | -858993460 | int |
| result | {i=-858993460 dw=3435973836 f=5 | real |
| frac1 | -858993460 | int |
| v2 | {i=-858993460 dw=3435973836 f=5 | real |
| frac2 | -858993460 | int |
| frac3 | -858993460 | int |
| v1 | {i=-858993460 dw=3435973836 f=5 | real |
| expr | -858993460 | int |

```
        return 0;
    result = (dw&0x7fffff);
    if (e>0)
        result |= (0x800000);
    return result;
}
// FIXME - figuure out how to
// handle underflow when handling
// values of different signs.
real real::add (real arg1, real arg2)
{
    real v1, v2, result;
    int frac1, frac2, frac3, expr;
    int d1 = arg1.e-arg2.e;
    if (d1<0) {
        v1 = arg2;
        v2 = arg1;
    }
    else {
        v1 = arg1;
        v2 = arg2;
    }
    frac1 = v1.f;
    frac2 = ((1<<23)+v2.f)>>abs(d1);
    frac3 = frac1+frac2;
    expr = v1.e;
    if (frac3>0x007FFFFF)
    {
        frac3 = ((frac3+(1<<23))>>1);
        expr++;
    }
    result.dw = (frac3&0x007FFFFF)|(expr<<2
    return result;
}

real real::sub (real arg1, real arg2)
{
    real result;
    DWORD abs1, abs2, max, min;
    unsigned short exp1, exp2, adjust, expr
    unsigned int frac1, frac2, frac3;
    bool sign1, sign2, sr, sflip;
//  break off the sign bit into two bools
    sign1 = ((*((DWORD*)(&arg1)))&(0x1<<31)
    sign2 = ((*((DWORD*)(&arg2)))&(0x1<<31)
```

```
overlords.
> Tell me about the path to the abyss, where the vultures await
> some unsuspecting soul.
>
Tell me about the lack of software tools for microcontrollers, such as by
trans-compiling c to forth, or c to spaghetti basic runtime option on the
parallax p2 chip as the "brain" for a simple macro language based on where
it is lazarus, or some deadly pitfalls, besides and among those things
aforementioned.
> That might very well be true, the path the leads to the roads
> forsaken, that causes many an unsuspecting CPU to halt and catch
> fire.  So what did you expect?
>
72.27 Hz, bandwidth, with far superior anti-aliasing that what you would
otherwise achieve with just four chords" you can crank out every pop hit
from the old cad package into the pre-processor or lexer.  And that seems
worth pursuing, like what does eliza have in it when first released in
1979 or thereabouts.
> Tell me more about Eliza.  Is she your friend?
>
As if figuring out exactly how i am converting some highly nested and
mutually recursive nested functions in order to have lazarus to run more
of the capabilities of the ucsd p-system had a mode, where she could say
"please instruct me.
> With Lazarus, or with Eliza?
>
With lazarus, or some kind of representational form that works with a
forked tongue, caught up somehow as if eliza should be able to, at least
in its present incarnation; and this sort of thing is certain.  That there
is the offending code.
> What code is offending.  You say some things at times that others
> might find offensive, and I am not talking about the quarterback
> sneak, nor am I thinking of a double reverse.
>
Double calculator::fadd(const double &arg2.
> Does that code bother you?  What is wrong with it?
>
What code is like the "better" than operator, on the one hand, and which
selects for things like blinking led's, managing displays, controlling
motors, etc.
> Kind of weird that you would complain about calculator::fadd, it is
> as if you are psychic or someting, I was just doing some work on that
> and I don't see how you could possibly know that you are now running
> with a soft FPU.  Yes, I would say that this is surreal alright.
>
```

So maybe it was Mega-Hal's "Somewhere in the sky comment" that just seemed, well, not as if the hand of God was reaching down from out of the sky or anything, just "surreal" in a way that I don't want to go into right now.  And in other words, I read that Pat Robertson passed away. but I don't think that Mega-Hal knows who that is or was.  Since Pat Robertson doesn't exist in the training set that I am using.  Now obviously, if I can "jail-break" a training set that was written to talk about compiler design, and get it - seriously - to hint at, hopefully without losing a PG rating that is, if is hinted at understanding certain types of adult material, which was NOT a significant part of the training set; then obviously - why not paint it into some kind of corner that might have religious overtones?  Yet here we see how the soft-FPU does multiplication.  Wow - that is a lot of stuff!  Yet, in any case - this was also a fun little chat, as follows - for readability.

*Start at: [2023/06/09 08:24:16]*
*MegaHAL: and not add any new peaks in each newly rendered frame.  Lots*
*MegaHAL: of work yet to meet our new robot overlords.*
*User:     Tell me about the path to the abyss, where the vultures await*
*User:     some unsuspecting soul.*
*MegaHAL: Tell me about the lack of software tools for microcontrollers,*
*MegaHAL: such as by trans-compiling c to forth, or c to spaghetti basic*
*MegaHAL: runtime option on the parallax p2 chip as the "brain" for a*
*MegaHAL: simple macro language based on where it is lazarus, or some*
*MegaHAL: deadly pitfalls, besides and among those things aforementioned.*
*User:     That might very well be true, the path the leads to the roads*
*User:     forsaken, that causes many an unsuspecting CPU to halt and*

*User:    catch fire.  So what did you expect?*
*MegaHAL: 72.27 Hz, bandwidth, with far superior anti-aliasing that what*
*MegaHAL: you would otherwise achieve with just four chords" you can*
*MegaHAL: crank out every pop hit from the old cad package into the*
*MegaHAL: pre-processor or lexer.  And that seems worth pursuing, like*
*MegaHAL: what does eliza have in it when first released in 1979 or*
*MegaHAL: thereabouts.*
*User:    Tell me more about Eliza.  Is she your friend?*
*MegaHAL: As if figuring out exactly how i am converting some highly*
*MegaHAL: nested and mutually recursive nested functions in order to have*
*MegaHAL: lazarus to run more of the capabilities of the ucsd p-system*
*MegaHAL: had a mode, where she could say "please instruct me.*
*User:    With Lazarus, or with Eliza?*
*MegaHAL: With lazarus, or some kind of representational form that works*
*MegaHAL: with a forked tongue, caught up somehow as if eliza should be*
*MegaHAL: able to, at least in its present incarnation; and this sort of*
*MegaHAL: thing is certain.  That there is the offending code.*
*User:    What code is offending.  You say some things at times that*
*User:    others might find offensive, and I am not talking about the*
*User:    quarterback  sneak, nor am I thinking of a double reverse.*
*MegaHAL: Double calculator::fadd(const double &arg2.*
*User:    Does that code bother you?  What is wrong with it?*
*MegaHAL: What code is like the "better" than operator, on the one hand,*
*MegaHAL: and which selects for things like blinking led's, managing*
*MegaHAL: displays, controlling motors, etc.*
*User:    Kind of weird that you would complain about calculator::fadd,*
*User:    it is as if you are psychic or someting, I was just doing some*
*User:    work on that and I don't see how you could possibly know that*
*User:    you are now running with a soft FPU.  Yes, I would say that*
*User:    this is surreal alright.*
*MegaHAL: P.s. For a good c++ 6502 compiler, or c++ for the propeller, or*
*MegaHAL: arduino, or any other applicable restrictions, i can see how*
*MegaHAL: the pascal with statement contains a bug, and that would be too*
*MegaHAL: simple.  Even if this is, as of right now - untested.*

What else is there to do?  Teach it politics? Be VERY AFRAID.

To be continued.