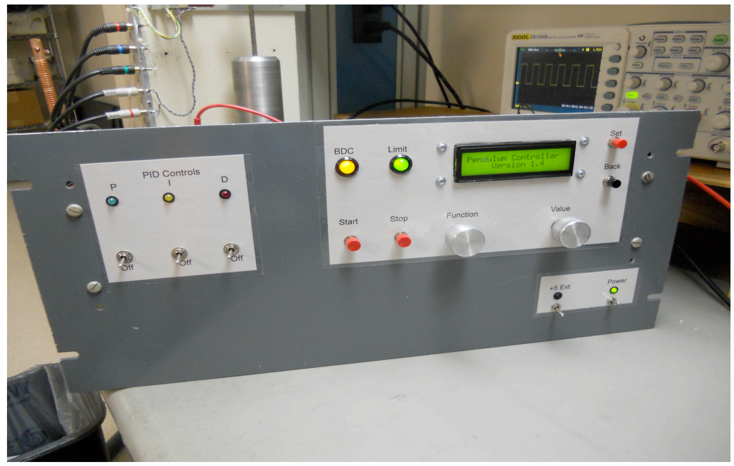
A Free Pendulum Controller and Analyzer

...the Adventure Continues

By Jim Hansen

In HSN issue 2011-5 I proposed a new method of electronically encouraging a free pendulum to obey Newtonian law in a novel (so I believe) fashion by maintaining pendulum velocity at the bottom of dead center (bdc). This is *not* controlling the pendulum. The methodology simply determines how fast it passes by and gives a gentle nudge of some computed amount to compensate for losses on the previous swing or swing averages.



At the end of that article I showed a collection of parts, including a massive surplus 16-inch rack panel to house a simple microprocessor, power supply, a collection of switches, LEDs and a 2-line LCD display.

This is a proof of concept vehicle and not intended as a "practical" pendulum clock controller. But strip off most of the switches and knobs, provide a reliable DC voltage source and this controller becomes one of unusual accuracy. As this is being written the electronics, the pendulum sensor, a basic magnetic drive and a good deal of the microprocessor code is operational and well underway. That is the upside. The downside is that, as always, feature creep has expanded the scope of this project.

Rather than a "simple" pendulum drive system that can drive a pendulum to the limits of its precision, it can analyze pendulum performance and provide any number of impulse schemes and strategies. A data logging feature sends data to a small data logger, from which the data can be easily uploaded to a pc, presumably running a spreadsheet program.

Another new "requirement" is some sort of mechanical pendulum impulse mechanism in addition to the magnetic drive system already under consideration. The reason for this is obvious to all clock persons – everyone knows that interesting things happen when the crutch bangs into the pendulum rod. Besides injecting a little energy into the pendulum, it also sets the rod "ringing" which, at the micro level, *can* cause aberrations in pendulum motion. This analyzer should be able to reveal these tiny effects through analysis of its drive power record.

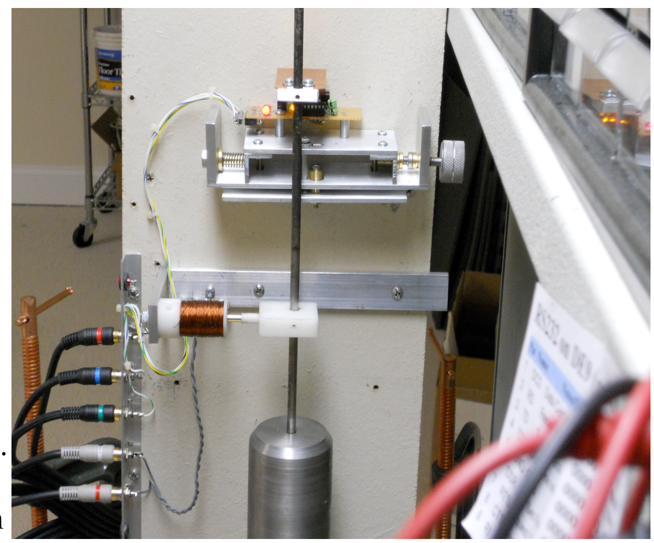
The analyzer can drive the pendulum bi-directionally as is common in most clocks, or uni-directionally such as in the case of the Hipp Toggle and Synchronome clocks, where impulsing swings in only one direction, every *n* number of swings. Simultaneously with this, it can also boost the pendulum to some amplitude limit, then let it coast down, then repeat the cycle similar to the way bang-bang systems operate. And it can *dynamically* adjust the impulse power level to maintain a constant pendulum velocity as measured at the bdc, using any of the fore-mentioned drive methodologies.

The ultimate controller presumes that we are working on pendulums of uncommon performance, and that ultimately the pendulum be driven in a finely tuned environment using the excitation scheme adopted during testing. Work continues on the basic Rev 1 version of this analyzer with about 1000 lines of C code written and debugged. It will start to be operational hopefully in a few more months.

Instrumentation
The sensor array used with the analyzer consists of two photo-interrupters. A word of caution to those who are new to these parts: do not hold them so the “ears” can be squeezed together. They are *very* weak in this direction and it doesn't take much force to bend them. Once bent, they often don't unbend, and if they do, they may break in half. Trust me this is so. I speak from some small experience in this matter.

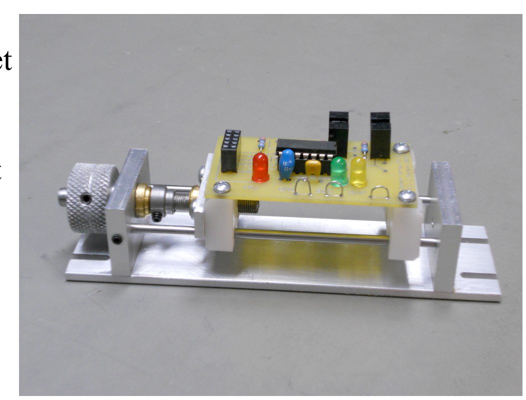
The analyzer measures pendulum velocity by passing a 0.020” slit over one of the photo-interrupters. Although this sounds easy enough, the size of the sensor aperture is very important and a common inexpensive photo-interrupter won’t do. The problem comes down to illumination – inexpensive interrupters are typically low-gain units and with a narrow slit there isn’t enough light passed through to properly actuate their detector amplifiers.

At first blush one might assume one photo-interrupter is about the same as the next, but not so when detecting slits. The issue is that the interrupter optical *sensor* diameter must be smaller, preferably a lot smaller, than the slit width. This is to ensure that the optical detector becomes fully illuminated as the slit passes by. If, for example, a photo-interrupter with a .030” diameter optical sensor were used, it could never be fully illuminated through a .020” slit. Its output can at times become erratic and unreliable.



Additionally, it is wise to use photo-interrupters whose outputs are “logic levels” and that have internal photo amplifiers and Schmidt trigger logic. This avoids a lot of analog logic design, testing and general fooling around. The Optek photo-interrupter that I’m using costs $6 and change each. So it is not cheap – nearly twice the cost of the microprocessor. But this is justified by reliable operation.

Photo 2 shows my first shot at a precision sensor array and basic (primitive) magnetic pendulum drive. Both are mounted behind the pendulum, and a flag with the slit attached to the pendulum passes through the sensors. This sensor array was a mechanical nightmare, overly complex, difficult to adjust and unstable. It had to go.



Also shown in photo 2 is the magnetic drive. It consists of a neodymium magnet mounted on the pendulum and partially set into the hollow core of a 1000-turn drive coil.

A word about coil philosophy, at least as far as I take it, is that the coil should be able to be placed across its power source indefinitely without damage. In this case, the 40-ohm coil is being driven from a 12V source and so the most current it can draw is 12/40 or 300mA. Safe enough, it stays nice and cool.

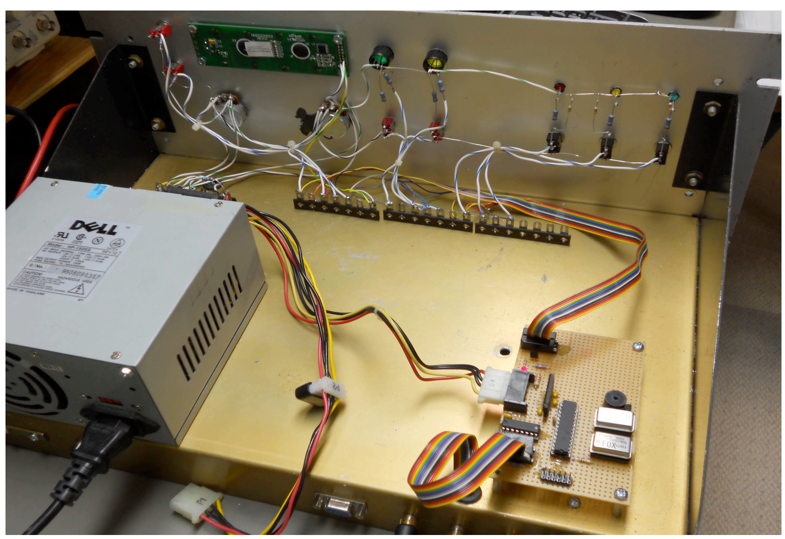
Photo 3 shows the Rev 2 version of the sensor array. It is now built around a printed circuit card which is used as a structural component. It mounts at right angles to the wall, thus somewhat simplifying the alignment requirement. The little wire loops shown along the front apron of the card are simple little test points, something for the 'scope probe to grab onto.

The LEDs on the sensor array aid in setting the mechanical alignment with the pendulum bdc. When adjusted properly and the pendulum is still, with the slit flag in place, the (green) bdc light somewhat flickers indicating good mechanical correlation with the pendulum bdc. The array adjustment uses a 40 tpi thread drive shaft, so the adjustment is pretty sensitive and non-ambiguous if your pendulum is (really) stationary. The other two leds are (red) +5v power applied, and (yellow) the amplitude limit indicator.

The pendulum has no way to be started once it is enclosed in a vacuum bottle. So the “Run” button on the control panel is provided to start the pendulum and thereafter keep track of which way it swings. The starting strategy is that after each impulse the controller waits a pendulum period until the pendulum swings back across bdc at which time another such impulse, this time in the opposite direction, is issued. This pattern continues until the amplitude sensor picks up the pendulum swing at which time the controller begins normal operation.

With the 12 lb pendulum bob shown, it only takes a few pulses at full power to get the pendulum swinging enough to flag the amplitude limit sensor. Since the controller knows which side the amplitude limit sensor is on, it can now identify the pendulum swing direction. In normal operation the bdc sensor output consists of pulses roughly of 25 msec duration when the pendulum is swinging about a half inch on either side of bdc.

So far, with my kludged setup, it seems apparent that for best pendulum stability, impulses should *pull* the pendulum rather than *push* it. My experience is that pushing usually results in slightly (and randomly) twisting the pendulum rod, complicating the magnet's mechanical compliance with the coil. Sometime later I'll work on a more sophisticated magnetic drive where hopefully this won't be an issue, but for now, impulses only pull, and it is readily apparent that today's magnetic fixturing is anything but precise.



Controller Description

The system is based on a Microchip PIC 16F24K22 microprocessor. This $3.57 chip provides a plethora of digital input-output lines, an analog-digital converter, gated timers, and more...even a 400-page user manual. For a processor of this complexity there is an aggressive learning curve that must be respected. Microchip supplies a free but competent C compiler, making complex programs easier to write. All code for this project is written in C.

Besides the microprocessor the hardware consists of a power supply removed from a junkyard pc, a front panel that holds all the indicator lights, switches, a two-line by 20 character lcd display, and interestingly, two potentiometers. These are used instead of mechanical switches to select functions and set “switch position” values by outputting a voltage that the microprocessor adc (analog to digital converter) reads. The program then performs whatever function is defined by that particular potentiometer setting.

The back apron has a number of phono jacks. These inexpensive connectors (also visible in photo 2) allow use of standard audio cables to connect to the pendulum sensor array and the pendulum excitation device, such as a coil for magnetic drive, or perhaps a crutch assembly for mechanically coupling to a pendulum or other coupling schemes. Also on the back panel is a DB-9 jack for connecting to a serial port of a data collection device or pc.

Photo 4 shows the back view of the controller. The microprocessor board is hand wired on standard prototyping perf board. Besides the 28-pin microprocessor, it has a 20mHz crystal oscillator for the system clock, a 1 mHz crystal oscillator for all timing functions, and a Maxim RS-232 driver chip for RS-232 output. An interesting motor-driver chip, a BA6886 (available from All Electronics), is used to drive all coil and pendulum excitation devices. It uses an H-bridge output driver and up to several amps of drive capability. An inexpensive speaker, used to produce “tick-tock” sounds, completes the electronic hardware. (At present they sound more like clicks than ticks, let alone tocks.)

As can be seen there is a seemingly never-ending number of wires running around the system, but that is all that there is - just wires. The microprocessor is simply connected to things without a lot of preamble. For example it can directly drive LEDs with only a current limiting resistor, or read switches while providing its own internal pull-up resistors. The board wiring is hand soldered 30 ga.wire wrap wire.

The annunciator for this system is a 2 line by 20 character display driven directly by one of the two serial ports on the processor. The second serial port drives the data logger, if used, or could be connected directly to a computer. In any event, the data sent out this port consists of the impulse record and, initially, the analyzer configuration, suitable for munching by Excel or other spreadsheet program.

Functionality

As feature creep slowly took hold of the project it soon became obvious that the original system functionality requirement was far too simple. All sorts of information is available when the pendulum is running, so the issue is how to make it visible. For example, instantaneous and continuous measurement of that mythological figure of goodness, “Q”, is a simple calculation, and “should” be easily displayable just by reading the impulse record. So naturally this feature should be added.

It also became apparent that the system could emulate the excitation patterns of just about any clock system with “just a little tweak here and there” in the code. To further the reality of the analysis, it is obvious that some sort of “noise” in terms of frequency, bandwidth, amplitude and amplitude roll off should be insertable into the excitation record. (I haven't figured out how to do that one yet.)

Another obvious “enhancement” - the mechanical excitation system - will eventually be desired, it being essentially a magnetically controlled crutch mechanism. At first this seemed like a big mechanical design issue, but then reason returned: this is a *pendulum* analyzer, not a *clock* analyzer. Thus it is not unreasonable to expect users to develop their own mechanics and mounting scheme for the magnetic crutch or other excitation assembly.

Controls

Looking at the control panel in photo 1, on the left are the three PID controller switches. When all are turned off, the system operates in a “conventional” clock mode, meaning that the excitation output is fixed at whatever impulse power is selected in the pendulum drive menu. When the “P” is turned on, the controller is a straight proportional system that can be programmed to any level in several ways. The idea here is that the drive power to the pendulum varies as the velocity error changes. In this mode, the larger the error, the greater the drive power. The I and D switches add finesse to the proportional system, making it a true PID controller.

Other controls include the start and stop pendulum control switches, the set and back switches that enter data or go back through the menuing, and the function and value potentiometers.

I elected to use a two-tier menuing scheme. Although quite fluid at the moment, the first tier is holds "category" information, the second manipulates variables under that category. For example, one category is called "Pendulum Dynamics" and under it are several choices, bdc averaged speed, instantaneous speed and variable controls of the PID functions. The "Drive Mode" category selects pendulum impulse strategies and power; Amplitude, Impulse and Data Logging options round out the top tier. Values for variables, such as "pendulum velocity", "pendulum length" or "# (number) of swings/impulse" are set by use of the value (right) potentiometer control with a digital readout shown on the LCD display; the desired value is entered into the program when the "set" switch is pressed.

Software development has reached the point where this mini-operating system has become the major point of focus and without doubt will end up taking more coding than the "real" work code of the system. It is readily apparent that after all the switches and other things used in the analyzer (but not in a clock) are stripped out, we're left with very little hardware and only about 20% of the code to make an accurate pendulum controller. This controller can then be buried somewhere in the case holding the pendulum.

I haven't included the schematic of the controller and sensor array because KICAD, the schematic editor I'm using, does not produce pdf files. But for those with an interest in the electronic side of things, a letter to our beloved editor will either get you the file once I've found a cheap and convenient way to pdf the drawings, or I can send a hard copy.

Note from the editor:

Jim has provided me the schematics in PDF form. I have placed them on the HSN website at >
www.hsn161.com/HSN/Hansen.pdf

Bob