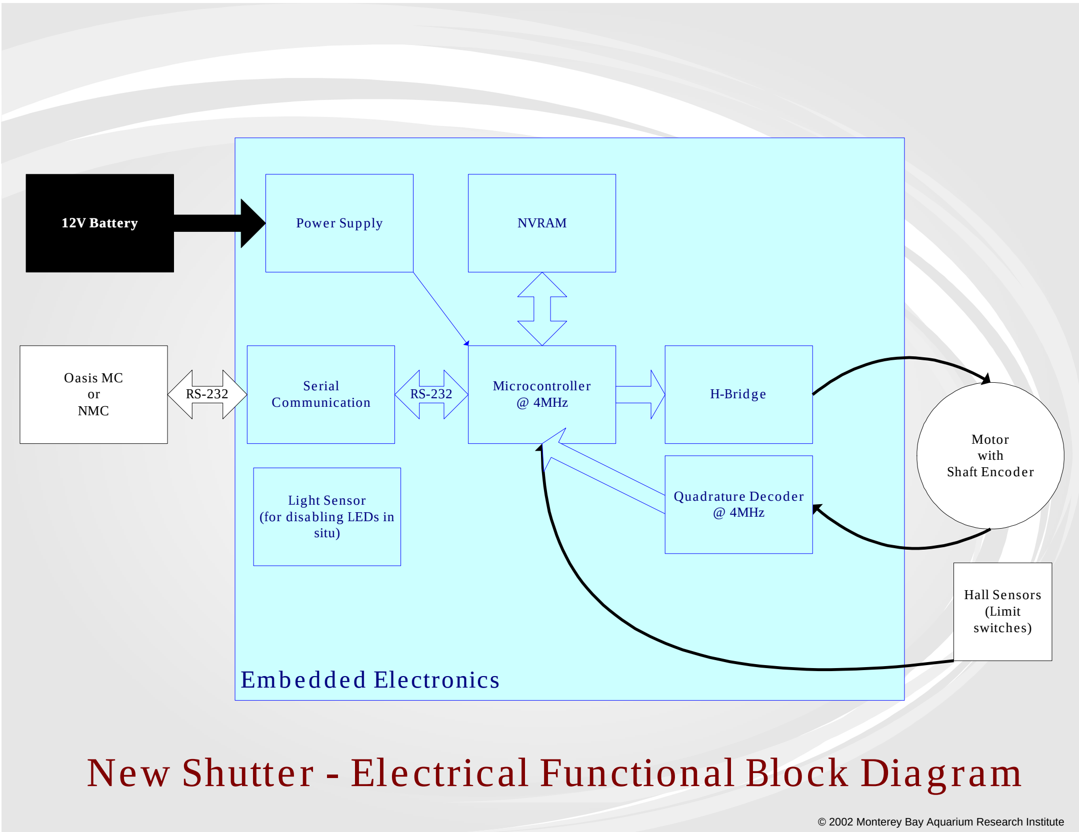
New Anti-biofouling Shutter
Mad Dash for Flash Cash Microchip Design Contest 2002
Entry #: M283

ABSTRACT/DOCUMENTATION:

The shutter was developed to reduce the effect of biofouling (barnacles, etc.) on a variety of optical instruments in use on oceanic moored platforms. The concept is to cover the optical windows of the optical instruments with a copper disk (a natural deterrent) when not making a measurement, thereby inhibiting the growth of fouling organisms on the window. The shutter is a system comprising two parts; the mechanism which contains the motor, gears, drive shaft feedback, drive shaft and copper plate; and the controller which consists of the motor drive electronics, the microprocessor, and the control software. The mechanism is attached to the side of the optical instrument and is submerged. The unit interfaces to a central processor via RS-232. The entire mooring runs on a typical lead-acid battery, so this shutter runs off the battery as well.

Figure 1. Biofouling on the M2 mooring, the number one problem for the moorings. The moorings are pulled in on a regular basis for maintenance: rebuild, cleaning and replacing instruments. Routine maintenance is also done in situ, with technicians boarding the mooring while out at sea.

The embedded electronics consist of a linear power supply, serial communications, an H-bridge, some non-volatile memory, a light sensor, a CPLD quadrature decoder and a microcontroller. Outside the main PCB, a motor with a shaft encoder drives the copper plate open and closed and two hall sensors, one at each end, operate a limit switches for initial calibration.



The functional design requirements were quite straight forward:

- ✓ The shutter must minimize bio fouling of it's sensor head.
- ✓ Electronics shall allow the motor to Open/Close the shutter
- ✓ The Open/Close process shall take no more than 40 seconds
- ✓ The Open/Close process shall not occur more often than every 10 minutes
- ✓ The shutter shall not occlude the optical window of the sensor
- ✓ The shutter must fit in the mooring elevator cage
- ✓ The shutter must survive a deployment duration of 1 year.
- ✓ The shutter must withstand a depth of 20m.
- ✓ The shutter must be compatible with both the old OASIS and new mooring controllers.
- ✓ The shutter should be controlled via a serial (either RS-232 or RS-485) 9600-8-N-1 data stream.
- ✓ The shutter should run off of a 12V battery (15V max.)
- ✓ The shutter must consume no more than 2A (nominally 0.5A).
- ✓ The shutter must be able to be power cycled while deployed.
- ✓ The shutter must run in a temperature range of 0^{o}F to 90^{o}F.

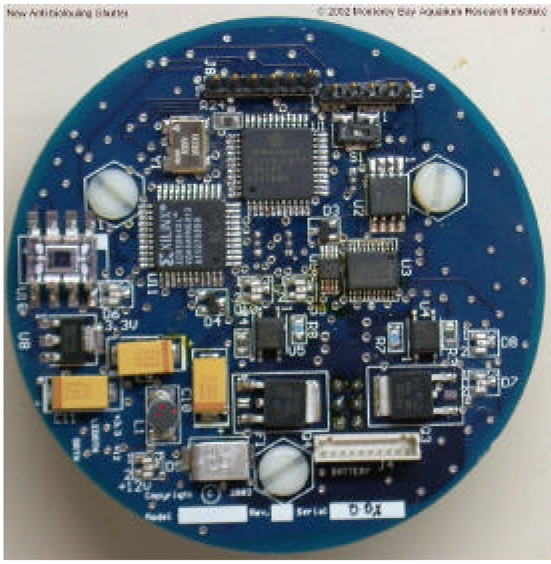
Figures 3a & 3b. a.) The shutter is a watertight housing composed of a base with a copper plate that opens and closes over an instrument's optical window. b.) Within the watertight housing, an embedded controller triggers a motor with a shaft encoder to drive the copper plate open and closed.

Beyond the basic requirements, I recognized that the unit would not see daylight 99% of it's life and I wanted status LEDs to indicate proper operation, so I added an automatic light switch to minimize power consumption - an important consideration since the entire buoy runs off of one solar-charged lead-acid battery for a year.

The first version of the shutter was just a simple motorized plate that opened when power was applied. It provided very little feedback to the mooring controller if something went wrong. At one point, this led to completely draining the buoy battery because the plate jammed on a zip-tie and the unit wasn't smart enough to shut down. To mitigate that problem, Mike added software that senses the velocity of the plate and reports an error if an unexplained halt occurs. The velocity is calculated by sampling the motor encoder every time the RTCC timer rolls over. If a minimum difference in motor position is not maintained between consecutive samples the motion is halted and a minimum velocity error is returned.

Since the unit would be submersed, we decided we needed a water sensor to detect if the housing flooded. If the unit catches the water fault in enough time, it sends an error back to the central processor, which can isolate the unit from the rest of the mooring.

Having worked with Microchip PICs before, Mike and I decided to use a flash based PIC for the unit's microcontroller. We chose the 16F877 because it is compatible with available in-circuit programming/debugging tools and it's low-power performance. It also offered enough I/O to meet



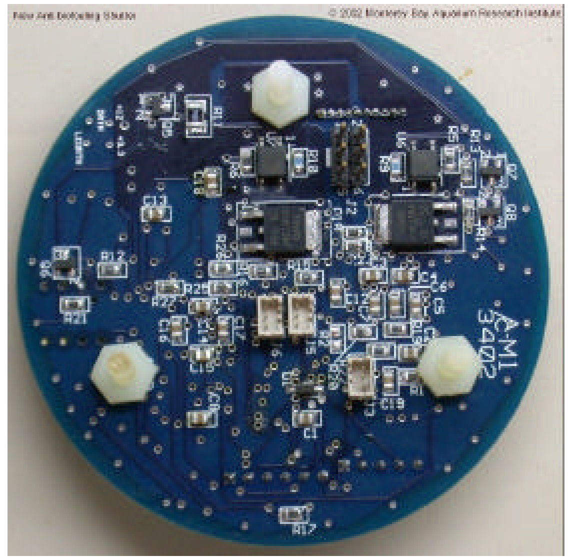


Figure 4 a & b. a.) The top side of the controller board for the new anti-biofouling shutter and b.) the bottom side of the controller board

the initial requirements and allow expansion if required. To minimize power consumption and make life interesting, I decided to pursue an all +3.3V design and I chose the PIC16LF877.

We encountered two problems as a result of pursuing an all +3.3V design. First, the in-circuit serial programmer/debugger (ICSP/D) used a +5V interface, so those lines required special consideration. Second, we originally wanted to use Working Technologies' bootloader, but that bootloader doesn't support low-voltage reprogramming so we decided to abandon the bootloader concept.

Rather than design a contraints specific solution for the motor control, we chose to implement a generic H-bridge and use a limit-switch/shaft encoder for position/velocity feedback. Since the device operates underwater, and a standard physical switch isn't watertight, we used a magnet/hall-sensor for the binary limit encoding. We used a Micronas unipolar hall sensor (HAL114).

Mike found a Xilinx application note for a quadrature decoder, so we implemented the decoder in a Xilinx Coolrunner CPLD (XCR3064XL-6-VQ44C) and ran the output into the parallel port on the PIC. The CPLD also contains the shoot-thru protection logic to prevent a state when both the top and bottom switches in the H-bridge are active.

I originally designed the H-bridge using all N-channel MOSFETs, but I was not happy with the predictable power consumption results. The final revision uses two P-channel and two N-channel MOSFETs and optoisolators to isolate the bridge from the high-voltage source. This structure allows any high-voltage source to be coupled to any logic level.

For manual operation of the device, it is powered from a 12V benchtop supply and connected to a PC's serial port using Hyperterminal at 9600-8N1. Upon power-up, the device issues a "RDY" status. In order to see what's being entered as a command the user types "echo<CR>" and the unit will now echo the commands. Two diagnostic commands "jog_o" and "jog_c" provide an easy way to move the copper plate into it's open and closed positions and pressing the space bar interrupts the movement.

For deployment, the device is homed using the command "set_stop". This allows the user to provide a 50% window or some other occluded view. Once the plate has been homed, the "open" and "close" commands are issued to open and close the shutter accordingly.

If there are any problems with the shutter during normal operation, like a water alarm, the device responds with "ERROR [error code]<CR> followed by a "RDY<CR>". Errors include water alarms, unknown commands, velocity mismatch, hall sensor limit or a number of parameter problems.

Even with the linear power supply, when it's running, not opening or closing, the unit consumes ~100mA, and it consumes ~160mA when it's opening and closing.

There are several things I'd like to do different for the next version of this unit. First, the Hirose connectors I used left a lot to be desired. The main 10-pin connector consistently had broken wires and the other connectors exhibited broken wires as well. This was partly due to the fact that they were assembled using solid rather than stranded wire, but even the stranded wire had a tendency to break over time. The pin diameter and spacing were also more designed for a 5mil PCB rather than the 7mil PCB that I wanted to use. Given the choice, I'd use a standard 0.100" connector and sacrifice the space.

Second, we could seriously reduce the nominal power of the device by implementing a sleep mode and a switching power supply. This would allow the device to constantly monitor the water detection circuit and provide immediate feedback to the central processor.

Third, I'd either implement a +5V design to allow the current version of Working Technologies' bootloader to work with the design, or I'd find another bootloader. We currently have to open the housing to reprogram the PIC and I'd prefer to keep the housing closed for any future updates. This would also allow code revisions to be forwarded from the central processor.