LLNL-MI-xxxxxx

Environmental Sample Processor (ESP)
Polymerase Chain Reaction (PCR) Module

DRAFT Requirements and Interface Specification DRAFT

Version 1.2

19 March 2009

Point of contact:
John M. Dzenitis, 925-422-6695, john.m.dzenitis@llnl.gov

Co-Investigators:
Vincent J. Riot, William J. (Bill) Benett, Dean R. Hadley,
Anthony J. (Tony) Makarewicz.

Lawrence Livermore National Laboratory

This work was performed under the auspices of the U.S. Department of Energy by University of California, Lawrence Livermore National Laboratory under Contract W-7405-Eng-48.

Funding is provided by the Monterey Bay Aquarium Research Institute (MBARI) from the National Aeronautics and Space Administration (NASA) project “An Environmental Sample Processor for Deep-Sea Seep and Hydrothermal Vent Applications” from the solicitation “Astrobiology Science and Technology for Exploring Planets” (ASTEP) in NRA 04-OSS-01: “Research Opportunities in Space Science.”

Concurrence sheet

ESP PCR Requirements Specification Version 1.2

Date: _____

Submitted by:

_____
John M. Dzenitis, Ph.D.
Project Lead
LLNL/Chemical and Biological Countermeasures Division

Approval:

_____
James M. Birch, Ph.D.
Engineering Group Leader, Instrument Development
MBARI

Executive summary

This document is the central repository both for specifications of requirements and for interface-control information for the Environmental Sample Processor (ESP) Polymerase Chain Reaction (PCR) Module. The PCR module is being developed by LLNL for MBARI in the NASA-funded Astrobiology Science and Technology for Exploring Planets (ASTEP) program. The main section is *2. Requirements specifications for the PCR module*, which documents the module that LLNL is developing for MBARI. This includes electrical, electronic, software, mechanical, thermal, and optical aspects. Another section is *4. Requirements for the ESP fluidics related to PCR*, which will help MBARI configure and operate the ESP to use the PCR module effectively.

Since this is the authoritative repository of requirements and interface information, any substantial changes to the document after release Version 1.0 need to be made through a change request and approval process.

Table of contents

Table of figures

Table of tables

Table of revisions

Table 1: Document revisions.

Date	Version	Major changes
20060329	0.6	First draft begun
20060928	1.0	First release version, resulting from PDR. Subsequent changes require documented change requests approved by both MBARI and LLNL.
20070306	1.1	New software commands
20090311	1.2	New software status commands (in Table 2) and number of data points (Section 2.7, Optical data).

1. Introduction

1.1. Document content

This document is the central repository for specifications of requirements and interface-control information for the Environmental Sample Processor (ESP) Polymerase Chain Reaction (PCR) Module. The main section is *2. Requirements specifications for the PCR module*, which documents the module that LLNL is developing for MBARI. Another section is *4. Requirements for the ESP fluidics related to PCR*, which will help MBARI configure and operate the ESP to use the PCR module effectively. To force clarity and make changes easier to track, information should appear in either one section or the other, not both.

This document will evolve continually up to the preliminary design review (PDR). After that point, the plan is to have an approved version of the document that is only changed by agreement of both LLNL and MBARI through documented change requests.

1.2. ESP and ASTEP background

The PCR module is being developed by LLNL for MBARI in the NASA-funded Astrobiology Science and Technology for Exploring Planets (ASTEP) program. NASA states, “ASTEP combines the science and technology communities in order to enable future space missions to determine whether life exists or has existed outside Earth. Through a detailed, collaborative analysis of Earth’s extreme environments, we can better prepare to understand analogous systems elsewhere.” ASTEP is part of the Office of Space Science, Solar System Exploration Division and the Mars Exploration Program.

One of the Earth’s extreme environments in consideration in ASTEP is the deep sea. MBARI has developed the Environmental Sample Processor (ESP), a remote electromechanical/fluidic instrument system that collects subsurface water samples, concentrates microorganisms (particulates) of various size classes, and automates application of molecular probes to identify microorganisms and their gene products. Under ASTEP funding, MBARI will further refine and deploy this instrument to evaluate the diversity and abundance of thermophilic and methanotrophic microorganisms present in deep-sea hydrothermal vents along the Juan de Fuca plate and methane seeps in the Santa Barbara Basin and Monterey Canyon. In addition, the team will enhance the analytical capabilities of the instrument by incorporating a nucleic acid purification and amplification module. Finally, MBARI will develop a detailed design for a modified ESP that is compatible with Martian surface operations and can search for astrobiologically relevant prebiotic and biotic compounds that may exist in dry, icy, or liquid samples. Refinement and deployment of the ESP as outlined here is relevant to science objectives found in the NASA Strategic Plan for Solar System Exploration.

LLNL is participating as a co-investigator on the project with responsibility for delivering the PCR module for the project.

1.3. Goals for PCR module project

The overall objective is to develop a polymerase chain reaction module for oceanographic microbiology in Monterey Bay Aquarium Research Institute's Environmental Sample Processor. This analytical module will be used in the ESP to perform *in-situ*, real-time genetic analyses in deep sea locations. The plan includes demonstrating real-time PCR in a prototype in January 2007, delivering four production modules to MBARI in June 2007, and providing technical support up to March 2008.

The general and specific goals for the module that feed into its requirements include

- Have performance similar to bench-top machines: appropriate thermal range, fast heating and cooling rates, four optical channels, good optical sensitivity, *et cetera*.
- Be usable in the ESP: small size, low power, moderate energy per reaction, compatible electronic interface, compatible fluidic interface, readily programmable, *et cetera*.

As part of this project, LLNL will provide guidance to help MBARI configure and operate the ESP to use the PCR module effectively.

1.4. PCR module scope definition

The PCR module is an "analytical module" in the terms of the ESP as a host platform. In typical use, the PCR module will receive electrical power, electronic commands, and a fluid sample from the ESP. The PCR module will execute real-time PCR thermal protocols, perform fluorescence measurements, and return optical results.

It is important to note that the PCR module does not contain any active fluidic functions (pumps, valves, *et cetera*) or reagents. These are provided by the host, and some of the requirements are detailed in the later section *Requirements for the ESP fluidics related to PCR*. The scope of the PCR module is diagramed in Figure 1 and shown in the context of the larger fluidics system in Figure 5 later in this document.

2. Requirements specifications for the PCR module

2.1. Voltage

The PCR module will operate on a nominal voltage of 10 to 16 V dc, unregulated but with little noise. Brief (< 20 ms) drops to 7 V dc are possible when other systems such as heaters come online. This requirement is determined by the battery source of the ESP and the desire to not have additional transformers. LLNL's current plan is to regulate the voltage to 9 V dc with 90 % efficiency on the heater power.

2.2. Current

The PCR module will draw a maximum of 5 A. This requirement is determined by the peak power that the ESP battery can deliver without a voltage drop that would impact other subsystems. There is a desire to have the module draw a maximum of 2 A, as this allows for the use of an Analytical Module power channel.

2.3. Energy

The PCR module will consume a maximum of 10 Wh (36 kJ) per 45-cycle reaction, including thermal cycling, fluorescence measurements, sending results, and decontaminating. This requirement is determined by the battery energy available to run 30 reactions over the course of a mission.

2.4. Data interface standard

The PCR module will communicate with the ESP using RS232 (EIA232F) protocols using 9-pin connectors. The ESP Host controller will be the Data Terminal Equipment (DTE) having a DB9-male connector and the PCR module will be the Data Circuit-terminating Equipment (DCE) having a DB9-female connector.

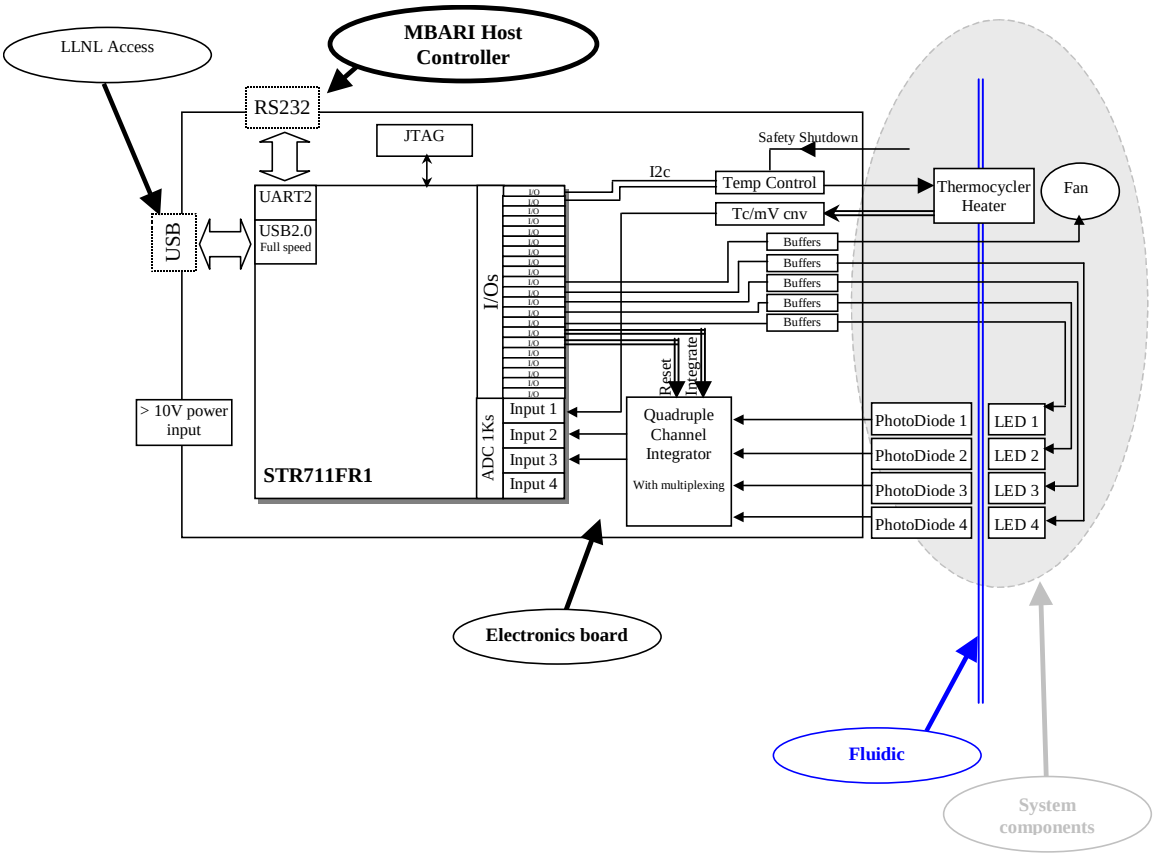

Figure 1: PCR Module and various interfaces.

Pin 1: Open
Pin 2: Received data (to DTE/ Host).
Pin 3: Transmitted data (from DTE/ Host).
Pin 4: Open
Pin 5: Signal ground.
Pin 6: Open
Pin 7: Request to send (from DTE/ Host).
Pin 8: Clear to send (to DTE/ Host).
Pin 9: Open

Other data interface specifications include 38.4 kbps baud rate, JTAG diagnostics/programming capability, RS232 hardware handshaking using the CTS and RTS lines, and ASCII human-readable commands.

These requirements are determined by the desire to have the PCR module readily incorporated as an Analytical Module in the ESP platform.

LLNL plans to also include a USB interface for the PCR module for LLNL use. I^2C capability has been discussed but was dropped as a requirement in our teleconference 4/14/2006. The present microcontroller candidate has built-in I^2C, and the PCR module board will have a connector to allow access to this function.

2.5. Message syntax

Here is a list of general specifications:

- Commands and queries will be terminated using the "line feed" or "carriage return" character. In the same manner, responses will be terminated by the "line feed" or "carriage return" character.
- The format will be ASCII-based in order to simplify debugging using a VT100 type terminal (e.g.: hyper-terminal in Windows).
- If the '?' character is at the end of the command, then the software will either return an acknowledgement or the data requested.
- If the '#<period in milliseconds>' sequence is at the end of the command, then the software will stream the data requested or return status if streaming is not available.
- If the command is not recognized, FAIL will be returned (followed by line feed) and buffer will be ready to accept new commands.
- The "Ctrl-U" character can be used to clear the module if the command buffer is suspected to be in an unknown state.
- When the module is first powered and ready to accept commands, a boot-up message will be sent. The syntax will be:

  @PCR MODULE:V<XX>.<XX><linefeed>

  where the <XX> values will be two digit numbers. For example: @PCR MODULE:V01.01

2.6. Commands and Responses

In this section, we first outline the types of commands and responses and then give a table with details of commands and responses.

2.6.1. Types of commands

The PCR module will accept the following types of commands from the Host.

1. Send your status
2. Load these configuration parameters (thermal PID, detection algorithm, thermal calibration)

3. Load this thermal protocol
4. Execute thermal protocol
5. Send stored data array
6. Stream optical, thermal, other data
7. Set optical state (LED on/off)
8. Set temperature
9. Send time
10. Send raw ADC optical data
11. Send current temperature in ADC or ºC
12. Abort
13. Reset
14. Shutdown

2.6.2. Types of responses

The PCR module will provide the following types of responses to the Host.

1. Syntax error (un-recognized command)
2. Ready for next command (acknowledgement of command)
3. Busy (and specify doing what?)
4. Execution or hardware error (and specify)
5. Current data array
6. Current temperature data (ADC or ºC)
7. Current optical data
8. Status of various components such as LEDs, fan, heater, *et cetera*
9. Stream of optical, thermal, other data

2.6.3. Details of commands and responses

The following table summarizes the RS232 commands. See also *3. Examples of PCR module protocols and their implementation*.

Table 2: Details of commands and responses.

Commands	**Descriptions**	**Answers**	**Notes**
RST? RST	Perform a soft reset of the system	*If '?' is present:* OK: once the command is performed FAIL: if error	Master command. Will not check if program is running
SHUTDOWN? SHUTDOWN	Initiate clean shutdown of the module	*If '?' is present:* OK: once the command is performed FAIL: if error or busy	Command type. Will not be executed if program is running (STATUS? returns RUNNING). If module is streaming, the ? is ignored and no status will be sent back, but the command will be executed.
ABORT? ABORT	Abort action that is causing the module to return anything but IDLE	*If '?' is present:* OK: once the command is performed FAIL: if error (module is shutting down)	Command type. Will be executed except if the module is already shutting down.
ECHO ON? ECHO ON ECHO OFF? ECHO OFF	Turns the echo on RS232 on or off	*If '?' is present:* OK: once the command is performed	Command type command. Will not be executed if program is running (STATUS? returns RUNNING) If module is streaming, the ? is ignored and no status will be sent back, but the command will be executed.
ECHO?	Return the status of the echo on RS232	*If '?' is present:* ON OFF	Status type command. Can be sent any time even when the program is running. If module is streaming, the command is ignored
STPSTREAM? STPSTREAM	Stop last streaming command	*If '?' is present:* OK: once the command is performed FAIL: if error (no streaming present)	Command type. Will be executed regardless of the status.

Commands	Descriptions	Answers	Notes
STATUS? STATUS#<msec>	Returns the status of the system. The system must be in IDLE mode before a program can be executed or any non-status command can be issued. If an error occurred, it will returned the last error message. The error message is reset after it is read.	IDLE RUNNING STANDBY ERROR:<Last Error Msg> If # is present, status will be streamed at regular intervals	Status type command If module is streaming, the command is ignored.
DIODE1 <intg>? DIODE1 <intg>#<msec> DIODE2 <intg>? DIODE2 <intg>#<msec> DIODE3 <intg>? DIODE3 <intg>#<msec> DIODE4 <intg>? DIODE4 <intg>#<msec>	Returns the ADC value corresponding to the photodiode for a given integration time defined by <intg> expressed in millisecond. If <intg> is too large it will be scaled down to the maximum available integration time.	<Integer> *0 to 4095* If # is present, diode readout will be streamed at regular intervals. It should not be streamed during program operation as it could cause some delays.	Status type command. Can be sent any time even when the program is running. There may be a delay in the answer if the module is running as the ADC needs to be shared. If module is streaming, the command is ignored.
LED1 ON? LED1 ON LED1 OFF? LED1 OFF LED2 ON? LED2 ON LED2 OFF? LED2 OFF LED3 ON? LED3 ON LED3 OFF? LED3 OFF LED4 ON? LED4 ON LED4 OFF? LED4 OFF LEDSTAT ON? LEDSTAT ON LEDSTAT OFF? LEDSTAT OFF	Turns the corresponding LED on or off. LED1, LED2, LED3 and LED4 correspond to the detector excitation LEDs. LEDSTAT is the status LED physically present on the control board	*If '?' is present:* OK: once the command is performed FAIL: if error	Command type command. Will not be executed if program is running (STATUS? returns RUNNING) If module is streaming, the ? is ignored and no status will be sent back, but the command will be executed.

Commands	Descriptions	Answers	Notes
LED1? LED2? LED3? LED4? LEDSTAT?	Returns the LED status. LED1, LED2, LED3 and LED4 correspond to the detector excitation LEDs. LEDSTAT is the status LED physically present on the control board	ON OFF	Status type command. Can be sent any time even when the program is running. If module is streaming, the command is ignored.
FAN ON? FAN ON FAN OFF? FAN OFF	Turns the fan on or off	*If '?' is present:* OK: once the command is performed FAIL: if error	Command type command. Will not be executed if program is running (STATUS? returns RUNNING) If module is streaming, the ? is ignored and no status will be sent back, but the command will be executed.
FAN?	Returns the fan status	ON OFF	Status type command. Can be sent any time even when the program is running. If module is streaming, the command is ignored.
HEATER RST? HEATER RST	Re-enable the heater after an over-temperature shutdown	*If '?' is present:* OK: once the command is performed FAIL: if error	Command type command. Will not be executed if program is running (STATUS? returns RUNNING) If module is streaming, the ? is ignored and no status will be sent back, but the command will be executed.
HEATER?	Returns the enable status for the heater	ON OFF	Status type command. Can be sent any time even when the program is running. If module is streaming, the command is ignored.

Commands	Descriptions	Answers	Notes
HEATSET <number>? HEATSET <number>	Set the heater value (in DAC values)	*If '?' is present:* OK: once the command is performed FAIL: if error	Command type command. Will not be executed if program is running (STATUS? returns RUNNING) or if the Temperature is set to an actual value. If module is streaming, the ? is ignored and no status will be sent back, but the command will be executed.
HEATSET?	Return the current heater setting in DAC values	<Integer> *0 to 4095*	Status type command. Can be sent any time even when the program is running. If module is streaming, the command is ignored.
TEMP <number>? TEMP <number>	Set the temperature PID loop setpoint to the specified number (temperature in ADC Values). If set to zero, the PID loop is disabled.	*If '?' is present:* OK: once the command is performed FAIL: if error or out of calibration range	Command type command. Will not be executed if program is running (STATUS? returns RUNNING) If module is streaming, the ? is ignored and no status will be sent back, but the command will be executed.
TEMP? TEMP#<msec>	Returns the current temperature in raw ADC Value	<Integer> *0 to 4095* If # is present, status will be streamed at regular intervals	Status type command. Can be sent any time even when the program is running. There may be a delay in the answer if the module is running as the ADC needs to be shared. If module is streaming, the command is ignored.

Commands	Descriptions	Answers	Notes
TEMPC <number>? TEMPC <number>	Set the temperature PID loop setpoint to the specified number (temperature in Celsius). If set to 0, the PID loop is disabled.	*If '?' is present:* OK: once the command is performed FAIL: if error or out of calibration range	Command type command. Will not be executed if program is running (STATUS? returns RUNNING) If module is streaming, the ? is ignored and no status will be sent back, but the command will be executed.
TEMPC? TEMPC#<msec>	Returns the current temperature in Celsius	<XX.XX> in Celsius If # is present, status will be streamed at regular intervals. Temperature returned can be negative if calibration is not appropriate.	Status type command. Can be sent any time even when the program is running. There may be a delay in the answer if the module is running as the ADC needs to be shared. If module is streaming, the command is ignored.
TIME? TIME#<msec>	Returns the value of the current timer counter	<Integer> current counter value of the timer If # is present, status will be streamed at regular intervals	Status type command. Can be sent any time even when the program is running. If module is streaming, the command is ignored.

Commands	Descriptions	Answers	Notes
PROGRAM <<number>:<command>;…;<<number>:<command>?	Set a program (PCR or any other type) in memory and check is the program has valid commands. The list of argument is defined as follows. Each instruction is numbered consecutively with a step number <number> then a colon and then the actual command. The following list summarizes all the available commands: TEMP,<Temp>,<time> This command set the temperature to the parameter <Temp> for the period of time defined by <time> TSTP,<Temp>,<time>, <loopincrement> This command set the temperature to the parameter <Temp> for the period of time defined by <time>. For each new execution (through loop or goto), the temperature is modified from the previous one by <loopincrement>. (continued…)	Here is the list of various answer. This command will always return a status. o OK the program is valid o ERROR:OVERFLOW the program has too many steps. o ERROR:LINE<num>: <error type> There is an error on step defined by <num>. The <error type> can be defined as follows: *Unknown Command* *Invalid parameter* o FAIL If streaming is running	Command type command. Will not be executed if program is running (STATUS? returns RUNNING) If module is streaming, the command will not be executed.

Commands	Descriptions	Answers	Notes
PROGRAM (…continued)	(…continued) READ,<integration1>,<integration2>,<integration3>,<integration4> This command performs a read with the integration time (ms) set by <integration(i)> on each four detector diodes and stores it in memory with the timestamp and current temperature. LOOP,<step>,<repeat> This commands loops back to the step number defined by <step> for <repeat> number of time. GOTO,<step> This command makes the program go to the step defined by <step> ENDP This command ends the program. It does not need to be present at the last step.		
STARTPROG? STARTPROG	Starts the current program. By default there is a program that just ends immediately as PROGRAM 1:ENDP would do.	*If '?' is present:* OK: once the program is started FAIL: if error	Command type command. Will not be executed if program is running (STATUS? returns RUNNING) If module is streaming, the ? is ignored and no status will be sent back, but the command will be executed. This command will put the status of the module into mode RUNNING

Commands	Descriptions	Answers	Notes
DATA?	Downloads data that has been stored during the last program	*<timestamp>:<diode1>,<diode2>,<diode3>,<diode4>,<temp>;…;<timestamp>:<diode1>,<diode2>,<diode3>,<diode4>,<temp>* List of ADC readouts for the various photodiodes with their appropriate timestamp and temperature. There should be as many values as there was READ commands executed during execution of the program.	Status type command. Can be sent any time even when the program is running. If module is streaming, the command is ignored
CALTEMP <a0>,<a1>,<a2>? CALTEMP <a0>,<a1>,<a2>	Set the polynomial coefficient for the temperature calibration. We should have the following equivalence: Temp = a0 + a1*(ADC) +a2*(ADC*ADC)	*If '?' is present:* OK: once the command is performed FAIL: if error or out of range.	Command type command. Will not be executed if program is running (STATUS? returns RUNNING) If module is streaming, the ? is ignored and no status will be sent back, but the command will be executed as long as the streaming has not been initiated with TEMPC#
CALTEMP?		<a0>,<a1>,<a2> List of the calibration coefficients for computing temperatures	Status type command. Can be sent any time even when the program is running. If module is streaming, the command is ignored

Commands	Descriptions	Answers	Notes
PIDTEMP <Kp,Ki,Kd,Per,fanon,fanoff>? PIDTEMP <Kp,Ki,Kd,Per,fanon,fanoff >	Sets the PID temperature control loop parameters Kp: Proportional parameter Ki: integration parameter Kd: derivative parameter Per: Loop period in ms (currently rounded to the lowest 10ms step) Fanon: Difference used to turn the fan on in ADC values Fanoff: Difference used to turn the fan off in ADC values	*If '?' is present:* OK: once the command is performed FAIL: if error or out of range.	Command type command. Will not be executed if program is running (STATUS? returns RUNNING) If module is streaming, the ? is ignored and no status will be sent back, but the command will be executed.
PIDTEMP?	Returns the PID temperature control loop parameters	<Kp,Ki,Kd,Per,fanon,fanoff> List of PID loop parameters	
SERIAL?	Returns the unique board serial number	SN<XXXXXXXX> Where the Xs are digits. The serial number is padded with 0s so that it is always 8 digit long	Status type command. Can be sent any time even when the program is running. If module is streaming, the command is ignored
VERSION?	Returns the firmware version in the form of the startup message	@PCR MODULE:V<XX>.<XX> Where the Xs are digits	Status type command. Can be sent any time even when the program is running. If module is streaming, the command is ignored

The following error messages can be returned:

- ERROR:Cmd Overflow: the command line is too long and cannot be stored before being processed
- ERROR:Cmd Unknown: command is unknown or wrong syntax
- ERROR:Busy: Device is busy and command is not allowed to be executed

- ERROR:No Streaming: STPSTREAM cannot be executed as there is no current streaming
- ERROR:Hardware Error: command has been tried and there was a hardware failure
- ERROR:Invalid Stream Period :Stream period is invalid for a streaming command
- ERROR:Invalid argument : Command has been recognized but the following arguments are invalid (out of bound or improper format)

2.7. Optical data

The optical data will be based on the digitized output of the integrator. The integration time will be set through the real-time clock of the microcontroller and can be set by the software. There is a maximum of 2048 data points available for storage in a program for software version 4.04 and higher. Earlier software versions will support up to 256 data points.

2.8. Thermal data

The PCR module will hold an internal thermal calibration defined as a polynomial of degree N (N = 2 for the first prototype) translating ADC value to actual temperature. The ESP host can modify the calibration constants at any time. The PCR module at power-up will hold a default "safe" calibration to prevent gross errors but accurate values will need to be loaded each time.

2.9. Data modes

The PCR module will be capable of providing optical and thermal data in modes of current value (return a single reading using low-level commands), stored values (standard format from a PCR), and streaming values (e.g., Return readings at *x* Hz).

2.10. Detection algorithm

There is no requirement for the PCR module to have a detection algorithm, since MBARI plans on doing the processing in the host or remotely.

However, for flexibility in other uses, LLNL expects to give the PCR module the ability to call cycle thresholds (CTs) using a threshold method including baseline correction analogous to the Cepheid SmartCycler. The parameters for the algorithm are number of cycles, voltage threshold, voltages at each cycle, number of cycles to lag baseline calculation, minimum number of points in the window, and minimum cycle for starting the calculation (n_cyc (*e.g.*, 45), v_thresh (*e.g.*, 0.050), v(1)…v(n_cyc) (voltages at each cycle), n_lag (*e.g.*, 4), n_min_win (*e.g.*, 5), and n_min_cyc (*e.g.*, 5)).

2.11. Size

The PCR module will occupy a maximum total volume of 1900 cm3. This volume will be comprised of separable volumes of 1250 and 650 cm3, measuring approximately 100 X 100 X 125 mm and 115 X 75 X 75 mm (4 X 4 X 5 and 4.5 X 3 X 3 in.). The limitations on the separation distance are not known. This requirement is determined by the amount of space available in the ESP and the desire to have some flexibility in how to locate the volume.

2.12. Mass

The PCR module will have a mass less than 2,000 g. This requirement is not very stringent for the ESP.

2.13. Shock & Vibration

The sock and vibration section is only for guidance and should not be considered as a mandate for testing practices.

2.13.1. Transit drop

The PCR module will be able to withstand a shock of 40*g*s in transport and still operate normally. This requirement is determined by handling that the ESP may experience while being transported and deployed.

The ESP and all its subsystems, when properly crated for shipment, shall survive transit drop test per MIL-STD-810F Table 516.5-VI (see Table 3).

Table 3: Transit drop test (MIL-STD-810F Table 516.5-VI).

MIL-STD-810F
1 January 2000

TABLE 516.5-VI. Transit drop test.

Weight of Test Item & Case kg (lbs)	Largest Dimension, cm (in)	Notes	Height of Drop, h cm (in)	Number of Drops
Under 45.4 (100) Manpacked or man-portable	Under 91 (36)	A/	122 (48)	Drop on each face, edge and corner; total of 26 drops D/
	91 & over	A/	76 (30)	
45.4 - 90.8 (100 – 200) inclusive	Under 91	A/	76 (30)	Drop on each corner; total of eight drops
	91 & over	A/	61 (24)	
90.8-454 (200 – 1000) inclusive	Under 91	A/	61 (24)	
	91 – 152 (36 – 60)	B/	61 (24)	
	Over 152	B/	61 (24)	
Over 454	No limit	C/	46 (18)	Drop on each bottom edge. Drop on bottom face or skids; total of five drops

2.13.2. Transit vibration

The ESP and all its subsystems when properly crated for shipment shall survive MIL-STD-810F transit vibration category 4. 60 minutes per 1000 miles of travel per 516.4c-1, MIL-STD-810F Fig. 514.5C-I (see Figure 2)

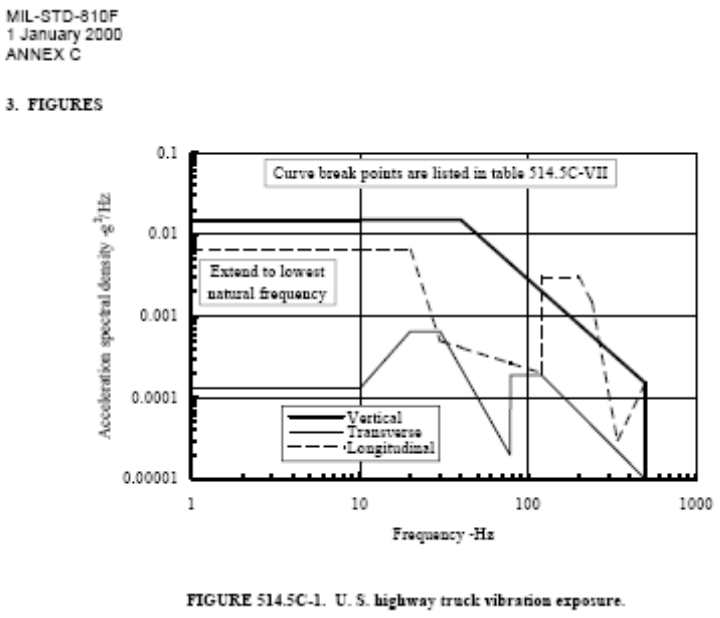

Figure 2: Transit vibration (MIL-STD-810F Fig. 514.5C-I).

2.13.3. Operational shock

The PCR module will be able to withstand a shock of 2*g*s while running and still operate normally. This requirement is derived from accelerometer readings from previous ESP deployments.

MIL-STD-810F Paragraph 516.5 Procedure I Functional Shock modified to account for shock cord isolation (see Figure 3).

5*g* peak acceleration, 15-23 msec, terminal peak saw tooth shock pulse, three shocks in each direction applied along 3 orthogonal axes (total 18 shocks).

MIL-STD-810F
1 January 2000

TABLE 516.5-I. Test shock response spectra for use if measured data are not available.

Test Category	Peak Acceleration (g's)	T_e (ms)	Cross-over Frequency (Hz)
Functional Test for Flight Equipment	20	15-23	45
Functional Test for Ground Equipment	40	15-23	45
Crash Hazard Test for Flight Equipment	40	15-23	45
Crash Hazard Test for Ground Equipment	75	8-13	80

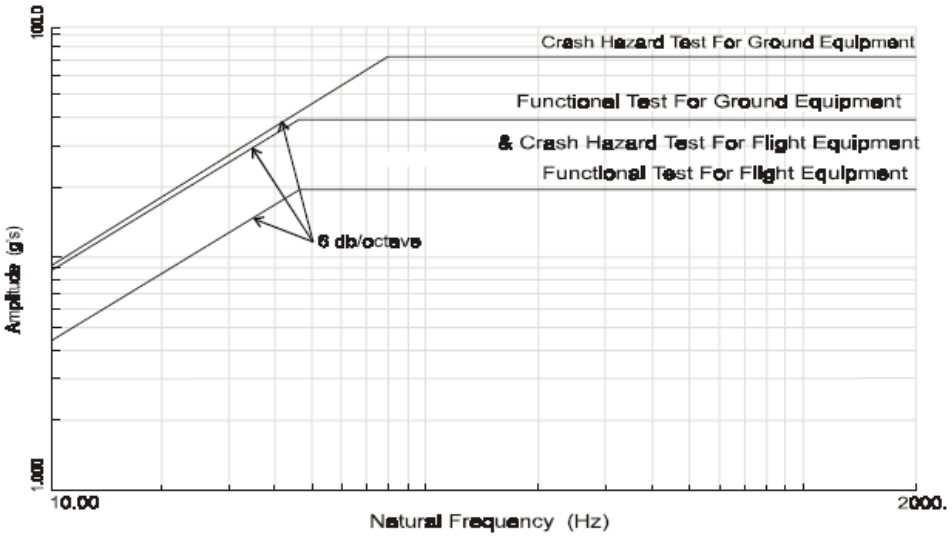

FIGURE 516.5-8. Test SRS for use if measured data are not available (for Procedure I - Functional Shock & Procedure V - Crash Hazard).

Figure 3: Functional shock (MIL-STD-810F Table 516.5-I and Fig. 516.5-8).

2.13.4. Operational vibration

MIL-STD-810F Minimum Integrity

60 Minute exposure along each axis per MIL-STD-810F Fig. 514.5C-17 (see Figure 4).

MIL-STD-810F
1 January 2000
ANNEX C

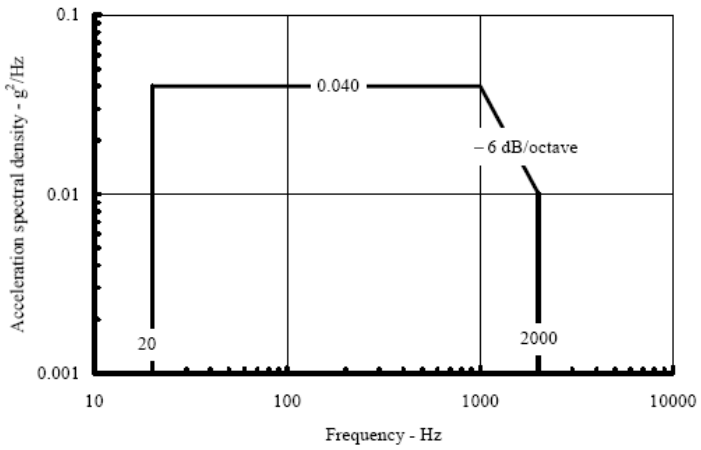

FIGURE 514.5C-17. General minimum integrity exposure.

Figure 4: Vibration minimum integrity (MIL-STD-810F Fig. 514.5C-17).

2.14. Orientation while running

The PCR module will be able to operate normally at angles up to 30 degrees from its normal vertical position. This requirement is determined by typical observed values at sea during operation.

2.15. Reaction temperatures

The PCR module will be able to hold reaction temperatures from 60 to 95 ºC with an accuracy of ± 1 ºC. This requirement is determined by the outer bounds of temperatures used for PCR and the likely sensitivity of the reaction to temperature control.

2.16. Heating rate

The PCR module will be able to heat from 60 to 95 ºC in 20 seconds with an overshoot of less than 2 ºC. This requirement is determined by the desire to be able to execute the APDS standard 60/72/95 45-cycle PCR protocol in 80 minutes.

2.17. Cooling rate

The PCR module will be able to cool from 95 to 60 ºC in 20 seconds with an undershoot of less than 2 ºC. This requirement is determined by the desire to be able to execute the APDS standard 60/72/95 45-cycle PCR protocol in 80 minutes.

2.18. Thermal environment

The PCR module will be able to operate in an ambient environment from 2 to 35 ºC. This requirement is determined by the missions that the ESP is required to undertake. For example we can see close to 0 ºC in the deep sea or in surf waters during the winter in some locations, and > 25 ºC in the surf waters of the tropics.

2.19. Humidity

The PCR module will be able to operate in an environment of 5 to 95 % non-condensing humidity. This requirement is determined by the conditions experienced in the sealed and deployed ESP.

2.20. Number of optical channels

The PCR module will have a minimum of 2 optical channels, and 4 channels are desired. The requirement for 2 channels is to enable a positive internal control along with a target signature in a reaction. The goal of 4 channels is determined by the desire to multiplex, to effect detection of more than 1 target in a given reaction.

2.21. Optical wavelengths

The PCR module will be configured to run excitation wavelengths centered at 480, 530, 575, and 635 nm all four with bandwidth 40 nm, and collection with respective long-pass filters 510, 550, 600, and 660, and respective emission filters centered at 525, 580, 620, and 680, all four with bandwidth 40. This requirement is determined by the dyes for TaqMan PCR that we have selected (FAM, TAMRA, Texas Red, and Cy5) for the goal of four channels in one reaction. Alternatives for FAM include SYBR Green, for TAMRA include Cy3, and for Texas Red include ROX. Other dyes and wavelengths can be used but will not be dynamically configurable; they will require change-out of light sources and optical filters.

2.22. Reaction volume

The PCR module will be capable of running reaction volumes from 20 to 40 uL. This requirement is determined on the lower end by the control capabilities of the fluidics system and the upper end by the desire to run sample volumes up to 10 uL.

2.23. Tubing outer diameter

The PCR module will use 1/16" OD fluidics tubing (0.062 +/- 0.001 inch or 1.59 +/- 0.02 mm). Other aspects of tubing (material, ID, *et cetera*) are covered in the section on Requirements for the ESP fluidics related to PCR.

2.24. Serial numbers

Each PCR module and replaceable heater module will have unique serial numbers. This requirement is determined by the desire to have the ability to track the history of these components and to maintain tables of calibration parameters, though it is acknowledged that those may change over time.

2.25. Fan reliability

If fans are used for cooling in the PCR module, the design will have a MTBF of 30,000 hours at L10 level (10 % of a lot fails) (or other means of specifying) traceable to the fan manufacturer's failure testing.

2.26. Documentation

A bill of materials will be provided for the PCR module and for the required associated fluidics.

3. Examples of PCR module protocols and their implementation

3.1. Introduction

Performing genetic tests in the ESP will require coordination between the ESP fluidics and the PCR module. Examples of this are given below. After this, PCR-module-specific protocols and their implementation in software are given.

1. Steps for setting up and performing a real-time PCR reaction
   1.1. Start power to the PCR module (ESP)
   1.2. Boot up from off state, receive configuration, and receive thermal protocol (PCR module)
   1.3. Build PCR plug (ESP fluidics)
   1.4. Move plug to module and valve off (ESP fluidics)
   1.5. Perform real-time PCR thermal cycling and fluorescence measurements (PCR module)
   1.6. Report results (PCR module)
   1.7. Decontaminate (ESP fluidics and PCR module)
   1.8. Shut down to standby (PCR module)
   1.9. Cut power to the PCR module (ESP)
2. Other actions
   2.1. Tune thermal parameters (ESP fluidics and PCR module)
   2.2. Tune plug positioning (ESP fluidics and PCR module)
   2.3. Check plug integrity/ positioning (ESP fluidics and PCR module)

The rest of this section contains examples of PCR-module-specific protocols and their implementation.

3.2. Real-time PCR

3.2.1. Description in English

These are the steps that the PCR module executes to perform the real-time PCR itself, after the reaction plug is already in place. (This is only an example case, and the parameters are set in the PROGRAM command.)

1. Hold temperature at 95 ºC for 75 s to hotstart Taq enzyme.
2. Hold temperature at 60 ºC for 30 s to anneal primers.
3. Read the fluorescence intensity on all channels.
4. Send results (The PCR module will have to be periodically polled by the host using the DATA command. The data is available at all times.)
5. Hold temperature at 72 ºC for 15 s to extend amplicons.
6. Hold temperature at 95 ºC for 15 s to separate amplicons.
7. Repeat 60/Read/72/95, 44 times.

8. Stop and turn off everything.

9. Send the final data (when the host uses the DATA command).

3.2.2. Description in the PCR module's programming language

`PROGRAM`

`0:TEMP,95,75000;1:TEMP,60,30000;2:READ,200,200,200,200;3:TEMP,72,15000;4:TEMP,95,15000;5:LOOP,2,44;6:ENDP?` (This example is one continuous line with no spaces except after PROGRAM.)

Returned value `OK`

`STARTPROG?`

Returned value `OK`

`DATA?`

Returned values (assuming channel 1 is agent and channel 4 is control)

`0:254,253,300,298,60;60:255,261,298,299,60;120:260,301,300,298,60;…;2640:3095,500,400,4120,60` (This example is one continuous line with no spaces. The ellipsis "…" represents similar data omitted for brevity.)

3.3. Melting curve

3.3.1. Description in English

1. Raise the temperature from 55 to 90 ºC by 0.2 ºC increments, reading and sending fluorescence intensity 2 s after reaching each hold temperature.

2. Send the data.

3.3.2. Description in the PCR module's programming language

`PROGRAM 0:TSTP,55,2000,0.2;1:READ,200,200,200,200;2:LOOP,1,175;3:ENDP?`

Returned value `OK`

`STARTPROG?`

Returned value `OK`

`DATA?`

Returned value

`0:254,253,300,298,55;2:255,261,298,299,55.2;4:260,301,300,298,55.4;…;350:3095,3598,3800,4120,90` (This example is one continuous line with no spaces. The ellipsis "…" represents similar data omitted for brevity.)

3.4. Thermal calibration

3.4.1. Description in English

Starting from scratch, one might use the following procedure for coarse calibration.

1. Set temperature to $T_{C,target,i}$. This will be based on the current calibration, which may be off.
2. Send ADC value $T_{ADC,i}$. User will be measuring $T_{C,actual\ i}$ by another method and record Celsius actual and ADC.
3. User will repeat above for as many points as needed for the calibration curve.
4. User will calculate and enter calibration constants a_0, a_1, a_2 for second-degree polynomial between T_C and T_{ADC}.

After the calibration is close, it is probably best to do a dynamic calibration by cycling and noting the temperatures of greatest interest, usually the 95 and 60 ºC. An example of this is shown below. One of the fine points to note is that the liquid plug may need to be occasionally refreshed (every 3 to 8 cycles, depending on the gassiness of the water) if bubbles are formed, because that will throw off the temperature measurement with an inserted thermocouple.

3.4.2. Description from a logbook

This is for a case where a linear fit is OK, which is typical. The a2 parameter is then zero.

CALTEMP? gave -29.5970,0.0366,0.0000.

Cycled (targeting 60, 72, 95) and got 56.8, 68.8, 92.0.

Recalibration:

Need a0_old (-29.5970), a1_old (0.0366), Tset1 (60), Tmeas1 (56.8), Tset2 (95),

Tmeas2 (92.0).

1. Back-calculate ADC1_old

ADC1_old = (Tset1-a0_old)/a1_old = 2448.

2. Calculate new slope a1_new

a1_new = a1_old*(T_meas2-T_meas1)/(T_set2-Tset1) = 0.0366*(92.0-56.8)/(95-60)

=0.0368

3. Calculate new offset a0_new

a0_new = Tmeas1 - a1_new*ADC1_old = 56.8 - 0.0368*2448 = -33.286.

Changed to CALTEMP -33.286,0.0368,0.0000.

Checked cycling and got 59.8, 72.0, 95.0.

3.4.3. Description in the PCR module's programming language

`TEMP 200?` (or, alternatively, `TEMPC 60?`) (for example)

Returned value `OK`

`TEMP?`

Returned value `200` (for example)

Read the actual temperature by other means and note the pair (200, temperature)

(Repeat for several temperatures and create a polynomial fit of degree 2 using least mean square best fit and get the three polynomial coefficients a0, a1, a2.)

`CALTEMP <a0>,<a1>,<a2>?`

Returned value `OK`

Note that the manual calibration may not have to be done frequently, but the accurate calibration values will have to be loaded by the host every time it powers-up the PCR module.

4. Requirements for the ESP fluidics related to PCR

4.1. Introduction

This is the place to put requirements for the ESP fluidics related to PCR that are not specifically addressed by specifying requirements for the PCR module, above. There should be no redundant information here, meaning no information should appear both here and above.

Figure 5 shows a fluidics diagram for implementing flow-through PCR. This implementation uses one pump and 3 14-port valves to achieve a choice of 2 sample sources, 2 master-mixes, 8 different multiplexed reactions, and DNA extraction

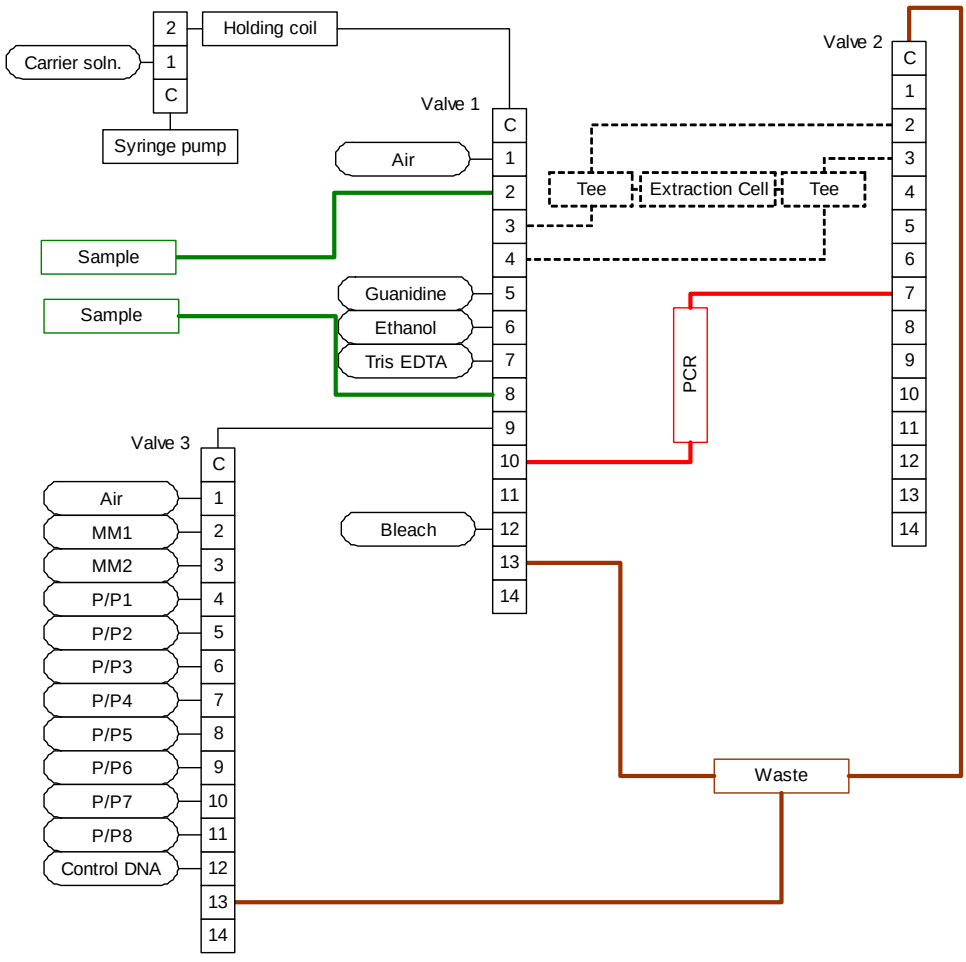

Figure 5: Flow-through PCR fluidics diagram using 3 valves to achieve a choice of 2 sample sources, 2 master-mixes, 8 different multiplexed reactions, and DNA extraction.

The PCR module itself is the block labeled “PCR”. There is an option to eliminate control DNA as a separate reagent by adding it to the probe/primer sets.

4.2. Reagents

The APDS baseline for reagents is

1. PCR
   1.1. Master Mix.
   1.2. Probe+primer sets, one per multiplexed assay, includes positive internal control probe+primers.
   1.3. Control DNA (could or should be added to probe+primer sets).
2. Decontamination
   2.1. Bleach (20% household bleach in water). Also used for decontamination of extraction.
   2.2. Water (carrier from pump is used). Also used for decontamination of extraction.
3. DNA solid-phase extraction
   3.1. Guanidine (adsorbing DNA)
   3.2. Ethanol (washing DNA)
   3.3. Tris buffer (eluting DNA)
4. Other
   4.1. Air.
   4.2. Sample.

4.3. Valves

The valve we use on the APDS is a 14 port Valco low-pressure multiposition valve, Part No. C25Z-31814-EMHX485, www.vici.com. This part number includes a microcontroller with RS-485 communication (RS-232 and CAN options are available). We suggest instead a slightly different version, Part No. C25Z-34814ECMTX485. This valve looks very similar, but there are significant differences. It has an integrated microcontroller. The rotor material is also harder, doesn't produce shavings, and stands up to dirt and high concentration salts better. This material requires a slower, higher torque motor. Contact Jack Reeves at technical support at Valco for the complete lowdown.

More details about fittings, *et cetera*, are available for MBARI (Commercial /Proprietary) in the Fluidics Bill of Materials (BOM). See, for example, fluidics_diag_astep20060724.xls.

4.4. Pumps

The pump we use on the APDS is a model number XP3000 from Tecan Systems, Part No. 725644 with a 1-mL syringe, Part No. 725591, www.tecan.com. The newest model syringe pump from Tecan Systems is the Xcalibur, Part No. 732734, which is the RS-485 option. This uses the same syringe.

On the low end, the ESP fluidics need to be able to meter 1 ± 0.5 uL of a reagent and move the PCR plug (5 uL air + 30 uL PCR + 5 uL air) a displacement of 300 ± 3 uL a flow rate of 1 uL/s.

More details about fittings, *et cetera*, are available for MBARI (Commercial /Proprietary) in the Fluidics Bill of Materials (BOM). See, for example, fluidics_diag_astep20060724.xls.

4.5. Tubing

The PCR module uses FEP tubing from Medical Extrusion Technologies with OD 0.062 +/- 0.001 inches and ID 0.045 +/- 0.001 inches. The rest of the tubing is PFA from Cole-Parmer. More details about sizes, part numbers, *et cetera*, are available for MBARI (Commercial /Proprietary) in the Fluidics Bill of Materials (BOM). See, for example, fluidics_diag_astep20060724.xls.

4.6. Fluidics protocols

The ESP fluidics and control need to be able to execute protocols like those below. For now, this is based on the APDS protocols, but that will be updated with protocols specific to the ESP.

4.6.1. APDS PCR reaction

1. Initialize Software – sets up write files, reads config, etc.
2. Plug Metering
   - 2.1. Initialize Pump – verifies pump in baseline position
   - 2.2. Prime Syringe – clears syringe and holding coil of bubbles
   - 2.3. Prime Thermal cycler – clears PCR tubing of bubbles and rinses with fresh PCR water
   - 2.4. Prime Reagent Reservoirs – guarantees fresh reagents from source vials and removes air bubbles from tubing
   - 2.5. Prime Sample Line – brings sample from sample source to valve
   - 2.6. Build Flow-Through PCR Plug
     - 2.6.1. Load 5 uL air
     - 2.6.2. Load 4 uL P/P
     - 2.6.3. Load 2 uL Control DNA
     - 2.6.4. Load 18 uL Master Mix
     - 2.6.5. Load 6 uL Sample
     - 2.6.6. Load 5 uL air
   - 2.7. Dispense Plug to PCR module – depends on length, materials, surface tension, moves 1 uL/s for approximately 300 s
3. PCR Initialize – sets software for the particular agent being run
4. Thermal cycling – conservative cycling parameters
   - 4.1. 75 seconds at 95 °C to hotstart Taq
   - 4.2. 30 seconds at 60 °C to anneal (also take optical measurement)
   - 4.3. 15 seconds at 72 °C to extend
   - 4.4. 15 seconds at 95 °C to melt
   - 4.5. Repeat 60/72/95 44 times

4.6.2. APDS PCR decontamination

1. Initialize Pump – verifies pump in baseline position
2. Dispense 10 uL Air to Thermal cycler – zone fluidics
3. Prime Bleach Line – brings bleach to valve
4. Dispense Bleach to Thermal cycler – fills PCR tubing with bleach, well past Thermal cycler
5. Dispense Air to Bleach Line – guarantees bleach can not leak into valve
6. Heat Thermal cycler to 95 for 120 seconds – bleach is most effective at high temperature
7. Rinse Thermal cycler 4 times
   - 7.1. Dispense Air to Thermal cycler – zone fluidics
   - 7.2. Prime Thermal cycler – simple rinse with PCR water

4.6.3. APDS DNA extraction

1. Prime guanidine, ethanol, and Tris lines
2. Prime sample line
3. Mix guanidine (100 uL) with sample (100 uL), dispense to pillar chip at 1 uL/s
4. Rinse pillar chip with ethanol (3000 uL, 100 uL/s)
5. Dispense air to pillar chip (200 uL, 10 uL/s to displace ethanol rinse solutions
6. Dispense Tris/EDTA elution buffer (10 uL, 1 uL/s) to pillar chip
7. Heat pillar chip to 80°C under PID control
8. Recover Tris eluent from pillar chip (10 uL, 1 uL/s)
9. Collect eluent in Eppendorf tube or dispense to flow-through PCR module
10. Turn off pillar chip heater

4.6.4. APDS DNA extraction decontamination

1. Dispense sodium hypochlorite solution to pillar chip (100 uL, 50 uL/s), sample inlet (100 uL, 50 uL/s) and sample outlet lines (100 uL, 50 uL/s)
2. Dispense DI H20 to pillar chip (1000 uL, 100 uL/s), sample inlet (1000 uL, 100 uL/s) and sample outlet lines (1000 uL, 100 uL/s)
3. Dispense air to pillar chip (150 uL, 10 uL/s), sample inlet (150 uL, 10 uL/s) and sample outlet lines (150 uL, 10 uL/s)

4.6.5. Check PCR plug integrity/ positioning

This has not yet been automated, but will be for the APDS and will be available for the ESP.