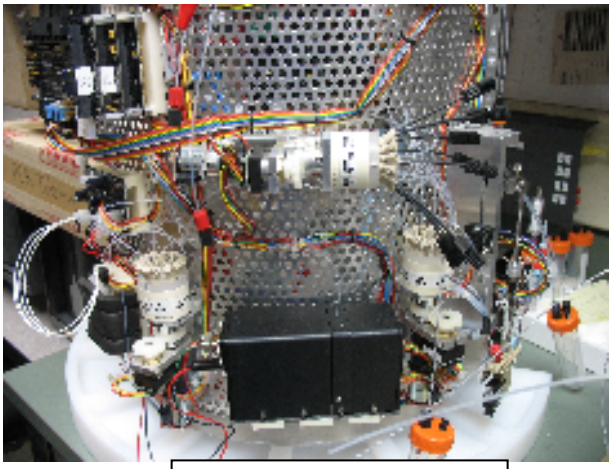
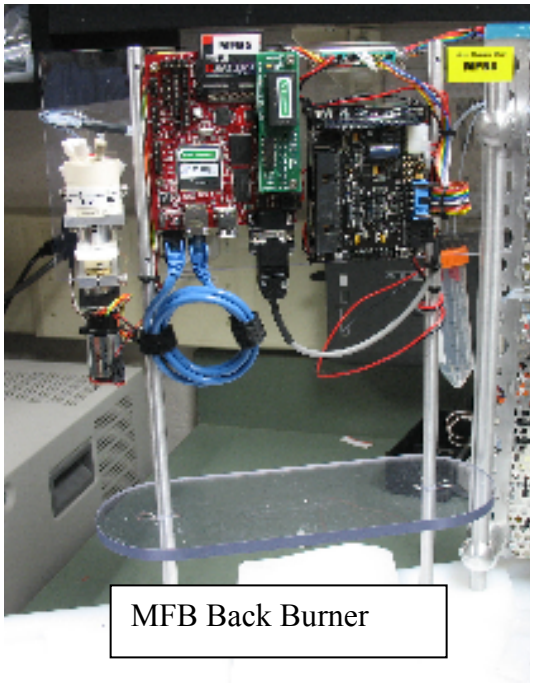
Preface:

The purpose of this manual is to acquaint new users to the ESP Micro Fluidic Block (MFB).



MFB Front Grille



MFB Back Burner

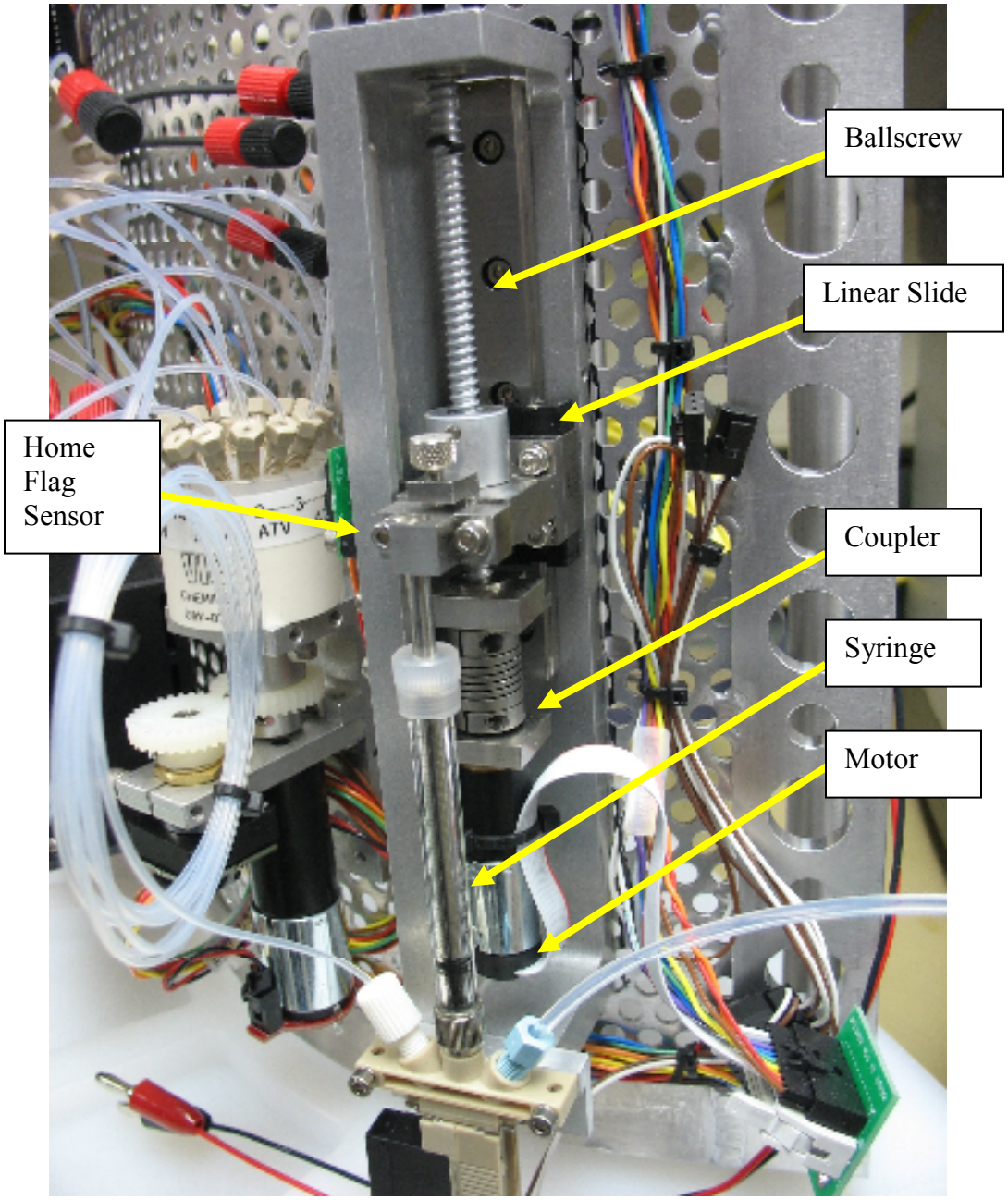
Table of Contents Page #

Table of Contents – continued Page #

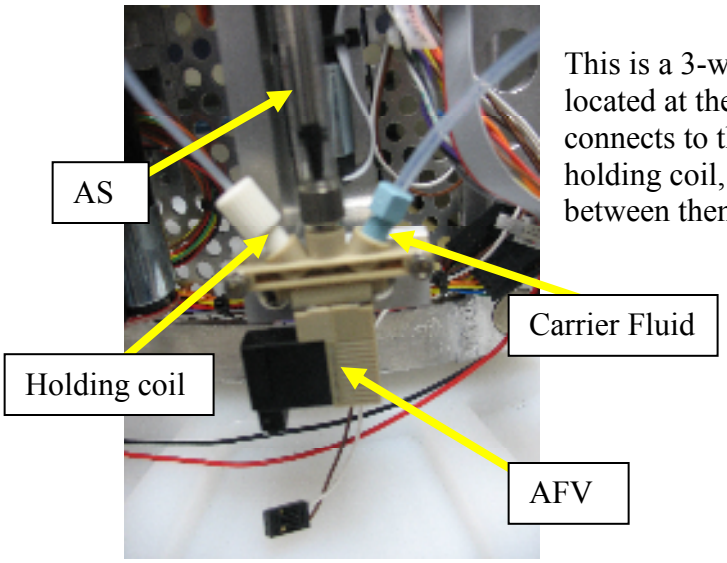
1. Parts Identification –
    a. Front Grille

i. <u>Analytical Syringe (AS) -</u>

The syringe assembly consists of a ballscrew, linear slide, coupler, syringe, motor, and home flag sensor. The syringe is a Zero Dead Volume 1.0 ml syringe from Klohn (PN #####). The motor is a Maxon (PN #######) 12V motor with a 512 count/rev encoder and a blah:1 gearhead.

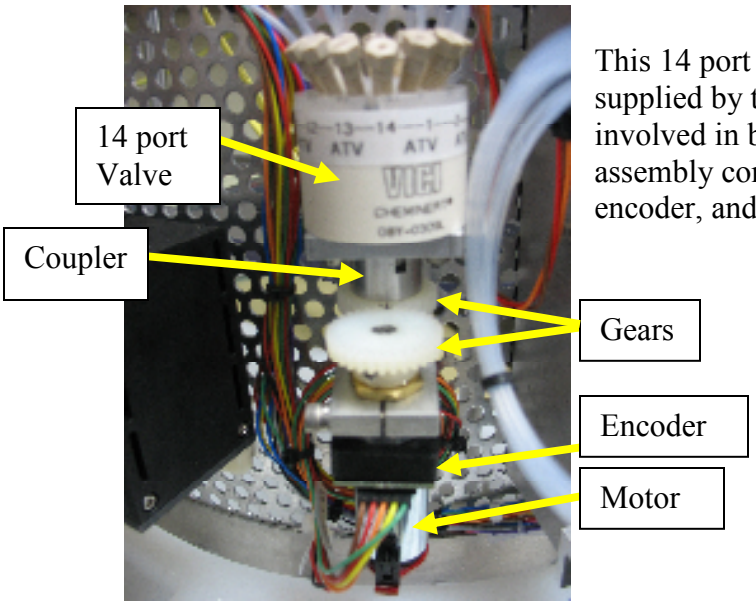


ii. Analytical Fluidics valve (AFV)–



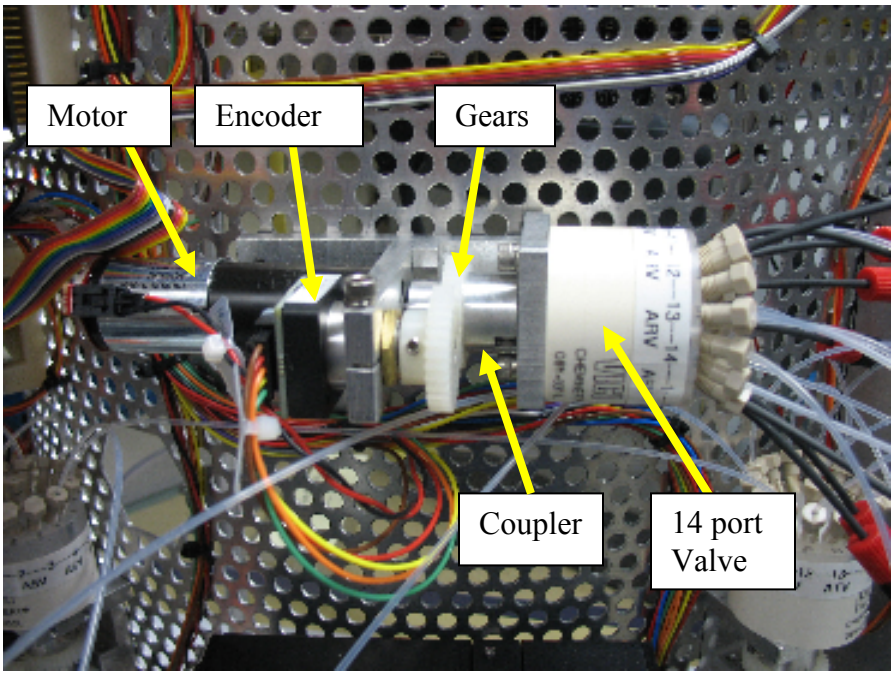
This is a 3-way flipper solenoid valve and manifold located at the bottom of the syringe. One side connects to the carrier fluid, the other to the holding coil, with the syringe being common between them.

iii. Analytical Top Valve (ATV) –



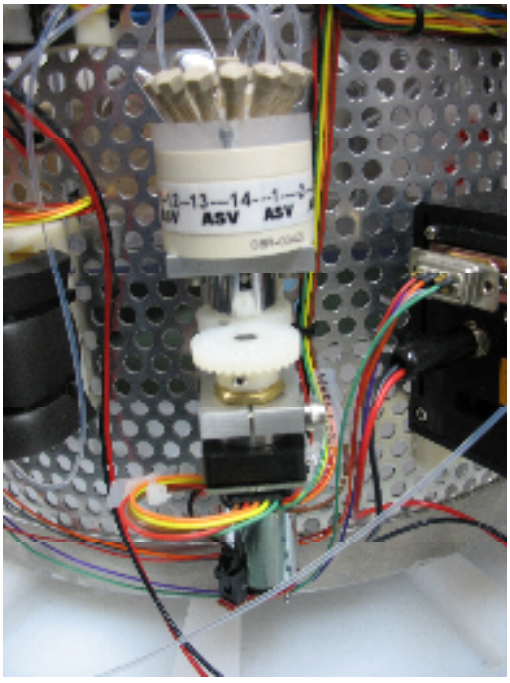
This 14 port stream select valve is center supplied by the holding coil and is mainly involved in building the reaction slugs. The assembly consists of a valve, motor, coupler, encoder, and a pair of matched gears.

iv. Analytical Reagent Valve (ARV) –



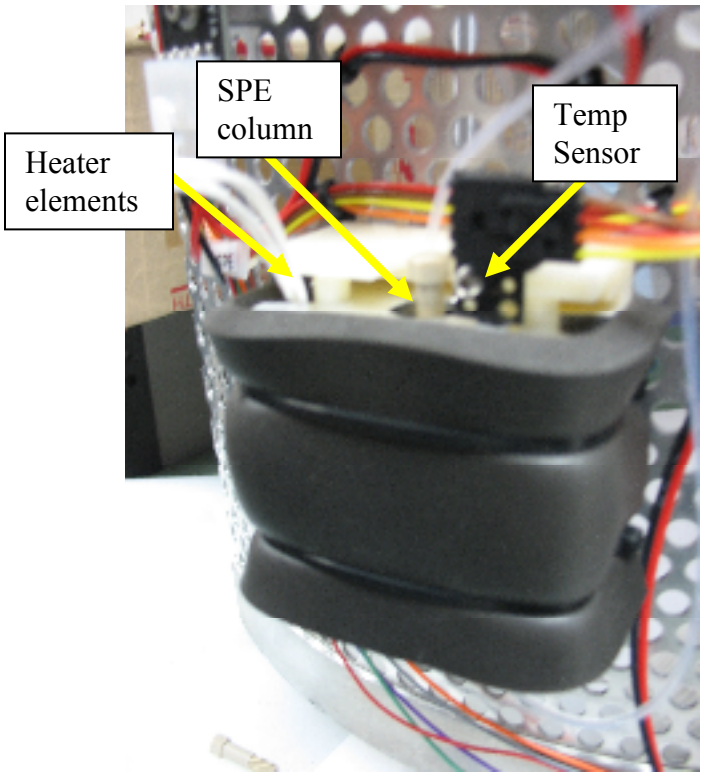
This 14 port stream select valve supplies the ATV and is mainly involved in supplying the various primers and master-mixes.

v. Analytical Sample Valve (ASV) -



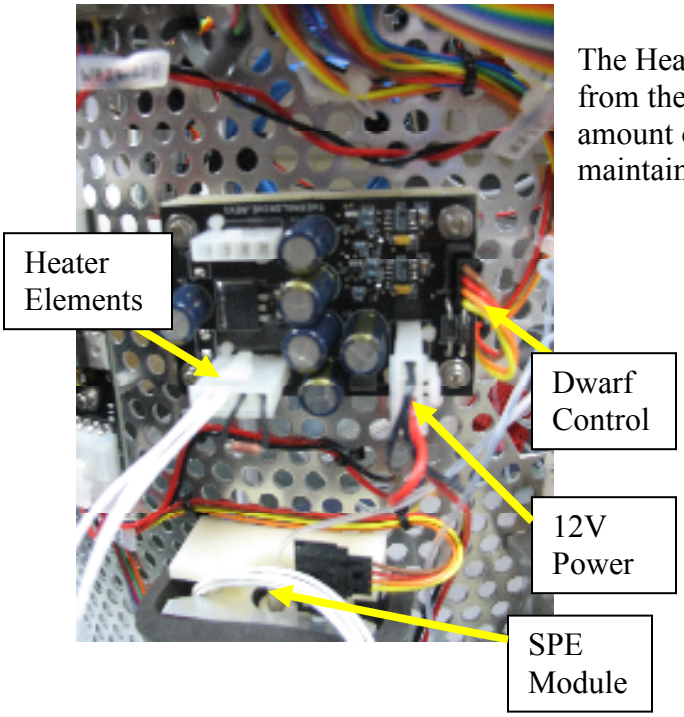
This 14 port stream select valve has a special stator that accesses the mixing coil and delivers the slugs to the SPE heater and Analytical Module.

vi. Solid Phase Extraction Heater (SPE) –



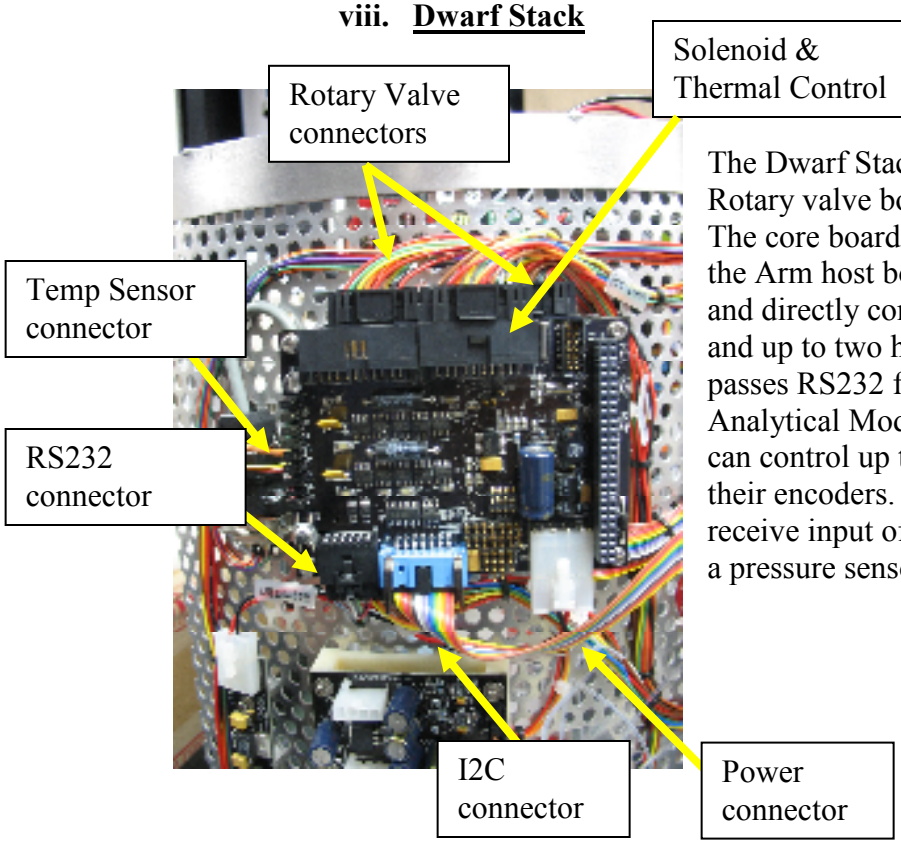
The solid phase extraction heater consists of an insulated aluminum block with thin-film Watlow heater elements on either side. The center of the block has a holder that contains replaceable extraction cartridges. There is also a temperature sensor potted into the block that is used for sensing.

vii. Heater Control Board -



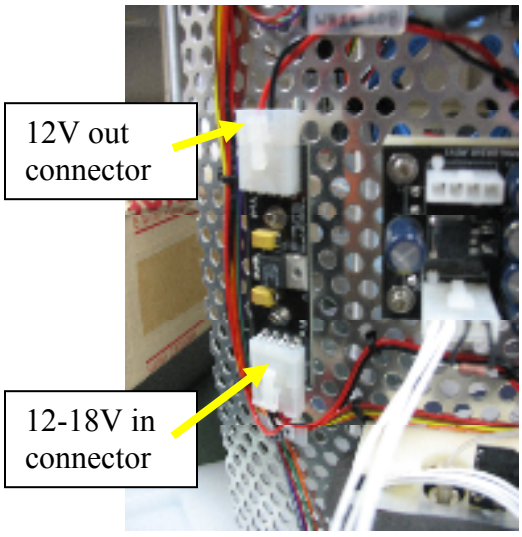
The Heater Control Board receives commands from the Dwarf Stack and supplies the correct amount of power to the SPE heater elements to maintain a desired temperature.

viii. Dwarf Stack



The Dwarf Stack consists of a 'Core' board, Rotary valve board and a Sensor board. The core board receives instructions from the Arm host board via the I2C network, and directly controls up to 8 solenoid valves and up to two heaters. The core board also passes RS232 from the Host to an Analytical Module. The Rotary valve board can control up to 4 DC brushed motors and their encoders. The sensor board can receive input of 1-2 temperature sensors and a pressure sensor.

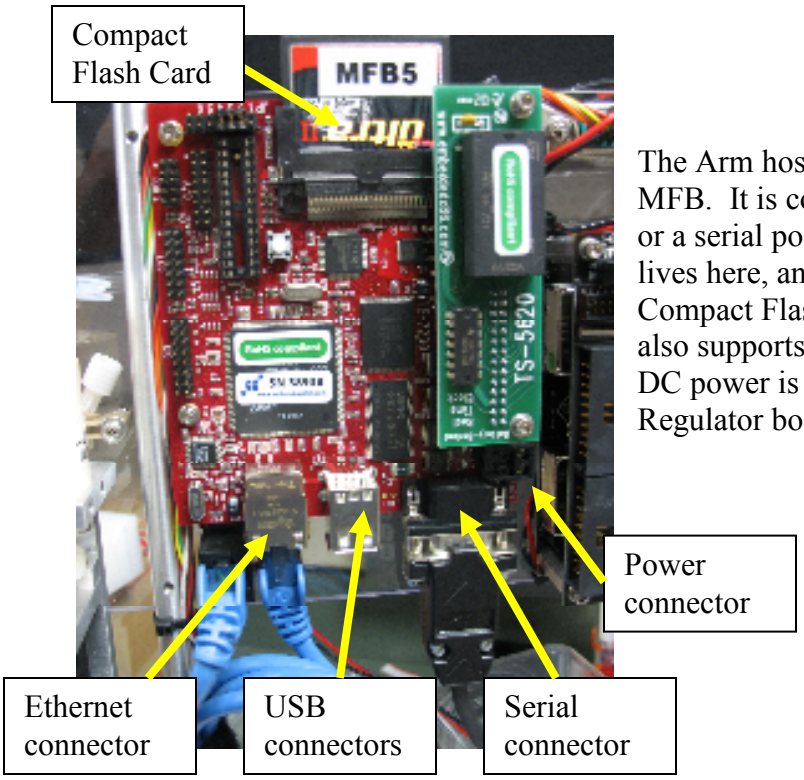
ix. 12V Regulator



The 12 Volt Regulator takes raw unregulated power in and supplies a clean 12.0 Volts out to the Dwarf stack. It also passes unregulated power to the Heater control board and to the Analytical Module.

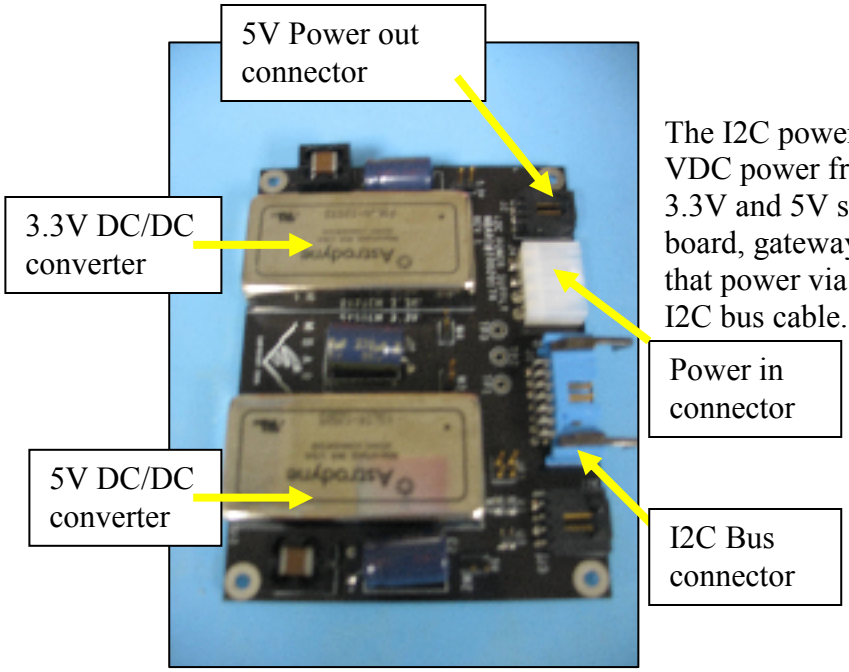
b. Parts Identification – Back Burner

i. ARM Host board



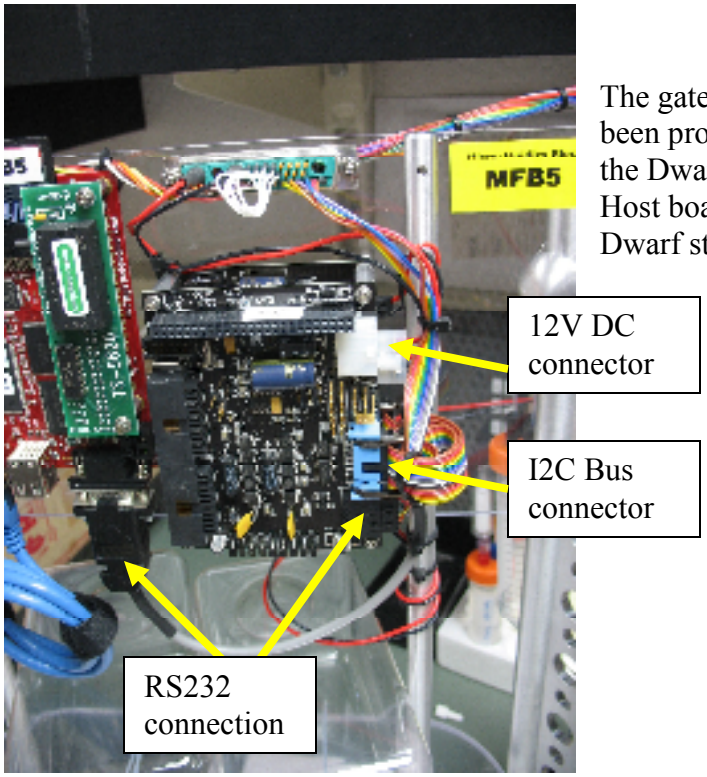
The Arm host board is the main 'brain' of the MFB. It is communicated with via ethernet or a serial port. The operating system, Linux, lives here, and there is an on-board 2GB Compact Flash for data storage. The board also supports two USB 1.1 peripheries. 5 V DC power is supplied via the I2C Power Regulator board.

ii. I2C Power Regulator board



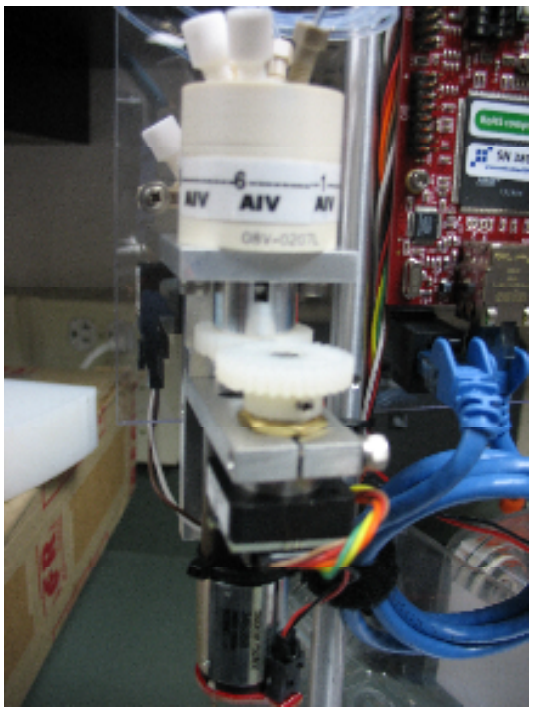
The I2C power regulator board receives 12-18 VDC power from a battery and conditions it for 3.3V and 5V system power needed by the Host board, gateway and the Dwarf stack. It supplies that power via a 14pin IDC connector that is the I2C bus cable.

iii. Dwarf Gateway



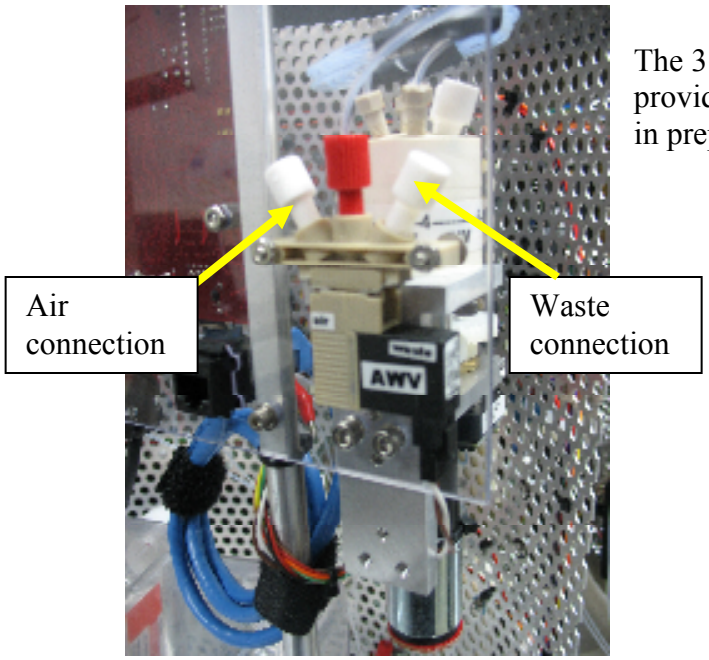
The gateway is a 'Core' board that has been programmed for use as a gateway to the Dwarf stack. It is connected to the Host board via a RS232 cable, and to the Dwarf stack via the I2C bus.

iv. Analytical Injection Valve (AIV)



This 6 port rotary valve provides a sample injection loop that can be manually filled, and then passed to the MFB. It is a two position valve, either fill the injection loop, or pass it to the MFB.

v. Analytical Waste Valve (AWV)



The 3 way flipper solenoid valve and manifold provide a path to flush and dry the injection loop in preparation for the next sample.

2. Electrical requirements
    a. Battery power supply

        A PowerSonic PS-12120 F2 12 Volt, 12 Amp.Hr. battery is used to provide un-interrupted power to the instrument.

    b. Battery charger

        A Xenotronix HPX-30 Sealed lead acid battery Charger is used to keep the battery at full capacity. It outputs 12 Volts, and 2 Amps.

3. Establishing communication to the instrument

This host computer is normally plugged into a wired Ethernet network via the RJ-45 jack on its front panel. It should integrate itself automatically into any network that supports both DHCP and DDNS.

If your network supports DHCP, once you have unpacked and visually inspected the MFB for damage, you should plug it into your network via a Cat 5 Ethernet cable before applying power to the MFB. When power is applied, a pair of tiny red and green light emitting diodes on the component side of the host computer board will blink briefly. One of the lights should remain solidly green if the board is working properly. If the lights continue to flash in a rapid, alternating pattern, the computer is failing to start. In this case, remove power and try again. If the problem persists, call MBARI.

Soon after the boot LEDs finish blinking, you should be able to observe two other blinking LEDs on the RJ-45 jack attached to the host computer board. Note that there are no LEDs on the MFB's Lucite panel. Follow the patch cable from that jack to the host board to see the network status LEDs. At least one of these should be illuminated and you should see irregular blinking corresponding to network activity. If these LEDs are still dark after 20 seconds, there is most likely a problem with the network connection. Check cables and routers. Try plugging a laptop or other computer into the same cable you intended to use for the MFB. If the other computer works, but the MFB shows no network lights, try another cable and/or try plugging directly into the RJ-45 connector on the MFB host computer, bypassing the patch cable to the Lucite panel.

If the network is plugged in when the MFB host is first powered on or reset, it should be ready to contact 30 seconds after you first observe network activity on its LEDs. However, if the network is plugged in after the host computer as started booting, the network integration process may take as long as five minutes. During this process it queries your network router for a unique IPv4

address and requests that it be associated with the unique name assigned to the MFB at the time of its manufacture. This "hostname" will be displayed on a prominent label affixed to the MFB's Lucite panel.

You can test whether the MFB is accessible a number of ways. In all the examples below the MFB's hostname is to be substituted for 'hostname' in the text:

1. Point a web browser on your network at the URL: FTP://*hostname*
2. Telnet to the host with the command: telnet *hostname*
3. Ping the hostname with the command: ping *hostname*
4. SSH into the host with the command: ssh mfb@*hostname*

Note that commands are to be entered on the command line of a computer on the same LAN. SSH is available under Mac OSx, Linux and Unix, but may not available under Microsoft Windows unless it had been installed as a 3rd party add on. The ping command on Linux and Unix will continue testing until aborted (typically with 'Control-C'). If telnet and ssh prompt for login or password you are ready to begin using the instrument.

If the above commands indicate that the MFB host computer is not reachable, it may be that your network supports DHCP but does not support DDNS. This means that the MFB can only be identified by its numeric IP address of the form 11.22.33.44 This numeric address is typically assigned by your network router box. Search your router's configuration web pages for information relating to DHCP assignment on your local network. You may find an entry for the MFB host name listed there. If you do, try substituting the IP address your router assigned in place of hostname in the above commands. If that works, you can identify the MFB host by this number. However, beware that it may change whenever the MFB is removed from the network or powered off.

4. Fluidic requirements and connections
    a. Carrier Solution
    b. Bleach Solution
    c. Ethanol Solution
    d. External source
    e. Reagent connections

5. Waste connections
    a. ARV waste container
    b. ASV/ATV Waste container
    c. AWV waste container
    d. Air Clean waste container

6. Controlling the instrument
    a. Open a new terminal

The MFB host computer supports two networking protocols to log in to it and issue Linux commands. The telnet protocol is insecure, but supported on just about every networked computer. The ssh protocol, or Secure SHell, is supported by all Linux, Unix and Mac OSx computers. Use ssh if your computer supports it.

Telnet always prompts for both a username and a password. SSH infers the user name from parameters input on the command line and prompts only for the password. Here's a typical command dialog for Telnet starting a telnet session: (typed text is in ***italics)***

> ***telnet MFB3***
Trying 134.89.10.238...
Connected to mfb3.shore.mbari.org.
Escape character is '^]'.

MBARI ESP Embedded Linux 1.7 -- 12/2/08 brent@mbari.org
MFB3 login: ***mfb***
Password: ***mclane***
Using esp2local, ESPhome=/home/mfb/esp2
mfb@MFB3:~$

Here's a typical command dialog for starting an ssh session:

> ***ssh mfb@MFB3***
The authenticity of host 'mfb3 (134.89.10.238)' can't be established.
RSA key fingerprint is
ae:05:da:5f:4a:05:a1:7b:b8:be:2b:64:4a:00:a4:06.
Are you sure you want to continue connecting (yes/no)? ***yes***
Warning: Permanently added 'mfb3,134.89.10.238' (RSA) to the list of known hosts.
mfb@mfb3's password: ***mclane***
Using esp2local, ESPhome=/home/mfb/esp2
mfb@MFB3:~$

Both the Telnet and SSH sessions function identically once started. One may have any reasonable number of either session type opened simultaneously.

To start the esp application that controls the MFB from the host's Linux command prompt:

esp@*hostname*:~$***esp***

Allow 30 seconds for the esp application to start and issue the esp command prompt:

```
@20:48:47.63 <MAIN> Copyright 2008 MBARI
/home/mfb/esp2/mfb/MFB3/configure.rb
$Revision: 1.1 $by $Author: brent $on $Date: 2008/12/06 05:47:00 $
@21:04:10.64 SPE.stop
@21:04:10.70 SPE.configure SPEconfig
@21:04:10.83 AS.coast
@21:04:10.87 AS.configure ASconfig
@21:04:11.05 PCR.pidtemp= 50,0.1,0,300,25,25
@21:04:11.15 PCR.caltemp= 4.803,0.0371,0
@21:04:11.19 All dwarves are running firmware version 2.59
ESPMFB3:001:0>
```

You should see messages indicating that various mechanical components of the MFB are being configured and initialized. If you see messages indicating that parts are "not responding" or "MISSING", execute the shutdown procedure described below and check cabling, etc., then power up, log in and try starting the esp application again.

The esp application is written in the programming language Ruby and its command prompt is a variant of Interactive Ruby or irb. It supports command auto-completion with the <TAB> key and command history recall using up and down arrow keys.

b. Ready the instrument - When the MFB is switched on, one must calibrate its Analytical Syringe and other components before using it. The command to do that is:
   - >***MFB.ready!*** (MFB.ready! will home the syringe and flush 1 ml of carrier fluid through the syringe and out the ATV waste port. The MFB is left with the syringe at empty, ATV at waste, ARV at waste and ASV at waste1.)
   - >***MFB.ready?*** This returns true if all axis are homed, false if not.

   - To quit the ESP application and return to the Linux command prompt:
     ESPMFB3:003:0> ***quit***
     mfb@MFB3$

c. Syringe movements
   - To initialize the home sensor
     >***AS.home.to :home***

   - To move to a absolute position on the syringe (eg .6ml)
     >***AS.to volume*** (eg AS.to .6)

   - To push a relative volume of fluid
     >***AS.push volume*** (in mL) (eg AS.push .01)

- To pull a relative volume of fluid
  >***AS.pull volume*** (in mL) (eg AS.pull .01)

- To stop the syringe
  Either >***control-C*** (if the last command was a syringe move)
  or >***AS.stop*** (if it is doing something else also)

- To release the syringe motor from holding a position
  >***AS.coast***

- To querry the raw encoder counts for calibration purposes
  >***AS.rawPosition***

- To change syringe speed
  > ***AS.reconfigure configurationname*** (eg AS.reconfigure ASslow1)

  Configuration names:
  ASslow0: maxSpeed = 200 counts/tic = 10 ul/s
  ASslow1: maxSpeed = 100 counts/tic = 5 ul/s
  ASslow2: maxSpeed = 20 counts/tic = 1ul/s
  ASconfig: maxSpeed = 350 counts/tick = 17.5 ul/s
  ASnorm: maxSpeed = 400 counts/tic = 20 ul/s
  ASfast: maxSpeed = 750 counts/tic = 37.5 ul/s

- To go a 'custom' speed
  >***AS.config.maxSpeed=17***
  >***AS.configure***

- To change syringe speed on the fly
  >***AS.to position, configurationname*** (eg. AS.to .5, ASfast)

- To fill the syringe full of carrier solution (1 ml)
  >***fillSyringeFull***

- To fill the syringe half full of carrier solution (0.5ml)
  >***fillSyringeHalf***

- To pull a volume a bleach from the ATV into the syringe.
  >***bleach(volume)***

- To safely pull a volume of ethanol
  >***ethanol(vol)***

d. Valve commands
   - There are six valves: ATV, ASV, ARV, AIV, AFV, and AWV

- To find out the names of the valve ports
  >***valvename.legend*** (the result is a list of encoderValues mapped to port numbers and names)

- To move to a port on any valve
  >***valvename.to portnumber*** (eg ATV.to 1) or
  >***valvename.to :protname*** (eg ATV.to :PCRwaste)

- To block off flow:
  >valvename.dialBetween portnumber, nextportnumber (eg ATV.dialBetween 1,2)

- To move the valve in a specific direction:
  >***valvename.to portnumber,:direction*** (eg ATV.to 8,:up) (eg ATV.to 8,:down)

- To get the encoder position of the Valve
  >***valvename.rawPosition*** (eg ATV.rawPosition)

- To rotate the valve by x ports
  >***valvename.advance x***
  >***valvename.retard x***

e. Pigtails commands
   - To clean the lines that connect to reagent coils on the ARV
     >***cleanPigTail(port)*** (eg. cleanPigTail(6) or cleanPigTail(:MM1))

   - To prime the reagent lines with new reagents
     >***pcrReagentPrime***

f. Slug building
   - To build a pcr slug, using the SPE Eluate, after running SPE
     >***pcrMission(template, fn=nil, options=nil)***

   - To build a pcr slug without using the mission script
     >***pcrSlug(template, mm, pp, endAir)***

   - To position a slug in the pcr module
     >positionSlug

   - To recover the slug after it has run in the PCR module
     >***pcrRecover***

   - To decontaminate the tubing after a slug has been built
     >***pcrDecon***

g. SPE commands
   - Set the SPE heater for a specified time to a temperature
     >***SPE.hold "h:m:s", temp***
   - Turn off the SPE heater
     >***SPE.stop***
   - To prepare a new SPE column for extraction
     >***columnBurnIn***
   - To wash out the solid phase extraction column
     >***washColumn***
   - To run solid phase extraction:
     >***runSPE***
   - To prime the Lysate and run SPE
     >***missionSPE***
   - To prime the Lysate line (done automatically in runSPE)
     >***primeLysateSA***
   - To recover the Eluate made in SPE
     >recover(vol) (eg. recover(.6) to recover 600ul)
   - To prime the eluate up to the ATV for building a pcr slug
     >***primeEluate***

h. PCR commands
   - To turn the PCR fan on/off
     >***PCR.fan = :on*** >***PCR.fan = :off***
   - To querry the PCR.state
     >***PCR.status***
   - To set PCR to a temperature
     >***PCR.temp = tempdesired***
   - To look at the PCR file for data
     >irb
     >***require 'instrument/pcr'***
     >***File.open('INSERT FILE NAME')*** {|file| Marshal.load file}
     (eg. File.open('09092008IRB.pcr') {|file| Marshal.load file})

i. Cleaning commands

- To clean the Holding Coil
  >***deconHoldingCoil***

- To flush water and bleach through the PCR and ATV-ARV line
  >***waterClean***

- To flush water through the PCR and ARV lines
  >***waterFlush***

- To clean the Eluate recovery line
  >***deconEluateRecoveryLine***

- To decon after Solid Phase Extraction
  >***deconSPE***

- To decon the Lysate line (done automatically in deconSPE)
  >***deconLysateSA***

- To wash and air out the Lysate Collection line
  >***deconCollection***

- To clean the PCR line when the tubing has been replaced or contaminated (This is a more thorough cleaning that concentrated on the PCR line)
  >***cleanPCRline***

- To clean the MFB, SPE, PCR, Lysate line and Eluate Recovery line after a run
  >***fullDecon***

7. Fluidic Diagram
8. Electrical Diagram