Commands as expected by the float controller via serial

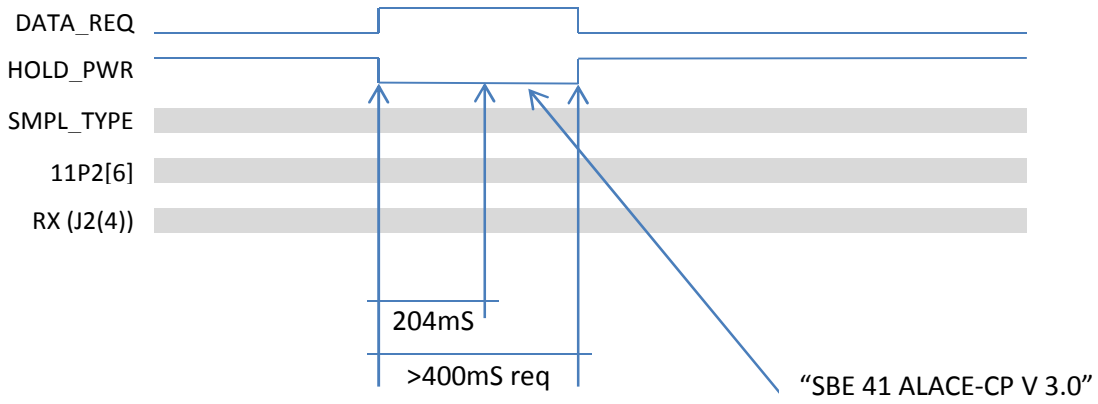
The float controller expects some commands and parses their responses for success and failure. There is some latitude in the response between the success and failure. In places the float controller pulls in strings and elsewhere it parses.

Command	success	fail	notes
\r	"S>"	timeout	
binaverage	"samples=" "done"	"bad:"	
outputpts=x	"S>"	timeout	x: 1,y,0,n
qsr	"powering down"		
startprofileN=x	"profile started"	timeout	x: ≥0, ≤120
stopprofile	"profile stopped"	timeout	
tswait=x	"S>"	timeout	x: ≥0, ≤255
dc	"S>"	timeout	
ds	"take sample wait time"	timeout	
dah	"upload complete"	timeout(s)	send up to 51200 characters

Commands as expected by the float controller via hardware interface

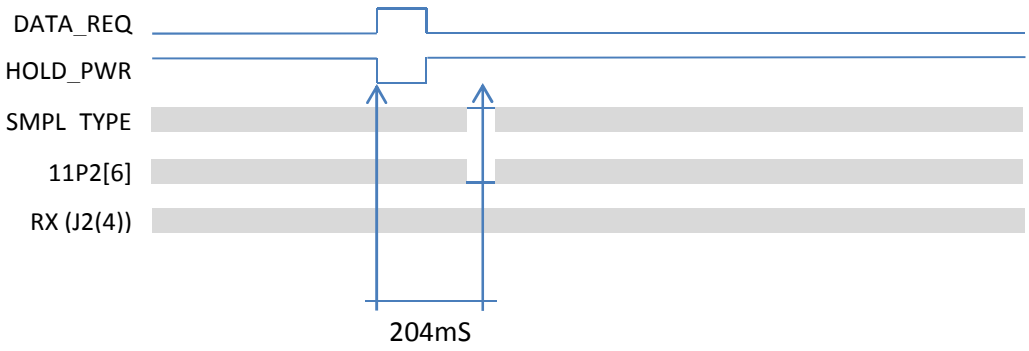
The DATA_REQ, SMPL_TYPE and RS232 RX lines are polled during startup and profiling. The timing on the original 41 is not precisely regulated with the decision for command or fast sample mode at ~204mS after the rising edge of DATA_REQ. DATA_REQ is also a source for wakeup. PONLATCH must be driven high within 500mS of a wakeup source (pulse or RS232)

Startup into command mode



If DATA_REQ is held high longer than the 204mS the states of SMPL_TYPE and RX do not matter and the instrument goes into command mode. Shortly after the 204mS scan the instrument ID string and firmware version is sent out the RS232 port. This string must be exactly the same in the float controller code for execution to proceed.

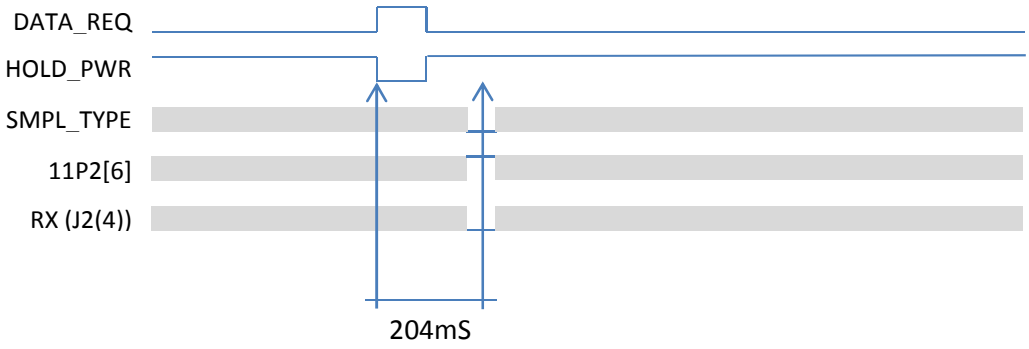
Pressure Temperature Salinity Sampling



DATA_REQ and must be de-asserted before 204mS and SMPL_TYPE must be asserted. Exact setup and holds should be measured through the interface circuitry. Multiple milliseconds of margin is the safest way.

The float controller expects a number formatted as "[ ]+(-?[0-9]{1,4}\.[0-9]{2}),[ ]+(-?[0-9]{1,2}\.[0-9]{4}),[ ]+(-?[0-9]{1,2}\.[0-9]{4})" This is a fairly pedantic string that expects " xxxx.xx, xx.xxxx, xx.xxxx". Note the leading space and spaces between units.

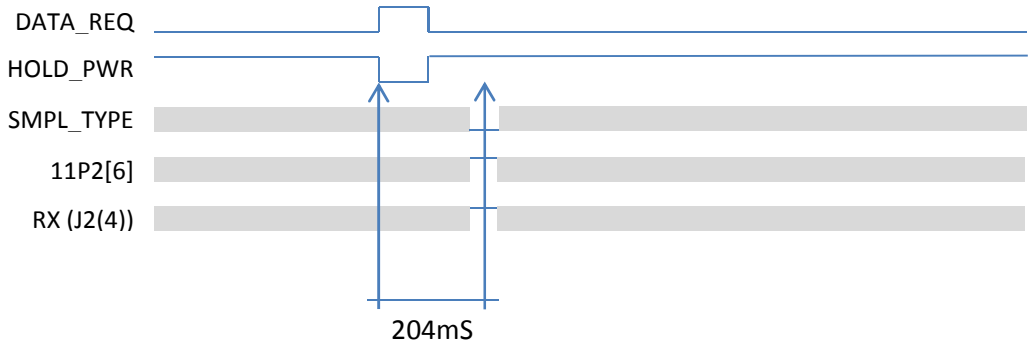
Pressure Sampling



DATA_REQ, SMPL_TYPE, and RX must be de-asserted before 204mS.

The float controller expects a number formatted as "^[^-+0-9.]*([-+0-9.]+)[^0-9]*" This is a string whose central section is numeric. "aabb12.33443bgs" passes while "12.33aa443" does not. The regex could be changed to "^[-+]*([0-9.]+)[0-9]*" to be extremely pedantic for floats.

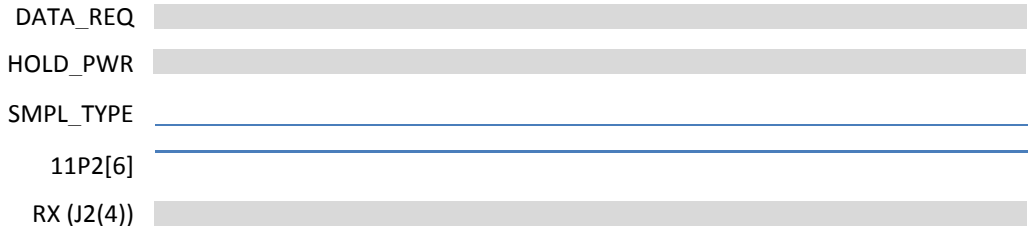
Pressure Temperature Sampling



DATA_REQ and SMPL_TYPE must be de-asserted and RX asserted before 204mS.

The float controller expects a number formatted as "[ ]+(-?[0-9]{1,4}\.[0-9]{2}),[ ]+(-?[0-9]{1,2}\.[0-9]{4})" This is a fairly pedantic string that expects " xxxx.xx, xx.xxxx". Note the leading space and spaces between units.

Pressure output during CP



Keep SMPL_TYPE low at all times during the measurement to enable the output of pressure sample data. The float controller strobes the SMPL_TYPE line (it calls it mode) for 100mS for every pressure request. This is unnecessary and a little dangerous as the pressure signal could be missed if the timing lines up just right.

Other commands

There are other serial commands that parameterize the instrument. Add as needed.

Command	EEPROM	notes	
prange	y	float	
psn	y	20 char string	
*serialnumber	y	word	
echocmd	y		

pcutoff	y	float (upper pressure cutoff)	
qs	n	sleep silently	
top_bin_interval	y	float	
top_bin_size	y	float	
top_bin_max	y	float	
middle_bin_interval	y	float	
middle_bin_size	y	float	
middle_bin_max	y	float	
bottom_bin_interval	y	float	
bottom_bin_size	y	float	
includetransitionbin	y	yes or no string	
ta0	y	float	temperature calibration
ta1	y	float	
ta2	y	float	
ta3	y	float	
tcaldate	y	25 character string	
cg	y	float	conductivity calibration
ch	y	float	
ci	y	float	
cj	y	float	
wbotc	y	float	
ctcor	y	float	
cpcor	y	float	
ccaldate	y	25 character string	
pa0	y	float	pressure calibration parameters
pa1	y	float	
pa2	y	float	
ptha0	y	float	
ptha1	y	float	
ptha2	y	float	
ptca0	y	float	
ptca1	y	float	
ptca2	y	float	
ptcb0	y	float	
ptcb1	y	float	
ptcb2	y	float	
poffset	y	float	
pcaldate	y	25 character string	
pumpon	n		
pumpoff	n		
tsr	n		

Serial Instrument Data

Float Controller Serial Data Sequencing

The float controller uses a pressure table to determine when a serial measurement should be taken during a profile. This measurement can include CTD data and is driven by GetObs in profile.c. The table is compiled into the firmware and spans 2000dBar to 1000dBar in 50dBar steps with the remaining steps to the surface split into smaller intervals. When the current pressure value of the table is exceeded by the pressure input from the CTD a sample is taken and the table number is incremented. Between samples the serial instrument is unpowered. This can be thought of as a stream of discrete observations (CTD + serial) separate from the continuous profile of the CTD.

Data Creation on the SBE 41

The SBE 41 almost immediately converts raw units to engineering and stores them as such.

Data Retrieval on the SBE 41

The float controller uses the command 'dah' or download average hex to retrieve a continuous profile. The creation of the binned data does not delete the original, computed data

Different firmwares have different conversion factors. The SBE 41CP UW V 2.0 code multiplies pressure by 10 and sends 4 hex characters to represent a pressure value.