APS Power Supply Controls

Claude W. Saunders, Oscar D. Despe
Controls Group / Power Supply Group , Accelerator Systems Division 3/31/94

87277
RECEIVED

FEB 14 1996

OSTI

Table of Contents

- 1.0 Introduction
 - 1.1 Components
 - 1.1.1 Power Supply Control Unit (PSCU)
 - 1.1.2 IOC
 - 1.1.3 Workstation
 - 1.2 Control Requirements
 - 1.2.1 DC
 - 1.2.2 Ramped
 - 1.2.3 Pulsed
 - 1.2.4 Hardware Synchronized Sets
 - 1.2.5 Knobs
- 2.0 Power Supply Control Units
 - 2.1 Introduction
 - 2.2 Development
 - 2.3 Code Overview
 - 2.4 Generating EPROMs
 - 2.5 Local Serial Control
 - 2.6 Remote Bitbus Control
- 3.0 IOC
 - 3.1 Process Variable Nomenclature
 - 3.2 Connecting a Database Record to Power Supply Hardware
 - 3.3 PAR/LTP/PTB Quad, Sext, Corr, Dipole Trim Records
 - 3.4 PAR Kicker Records
 - 3.5 PAR Septum Records
 - 3.6 Booster Temporary Quad, Sext, Dipole Records
 - 3.7 Booster Temporary Corrector Records
 - 3.8 Booster Kicker Records
 - 3.9 Booster Septum Records
- 4.0 Control Screens
 - 4.1 Summary Power Supply Status
 - 4.2 DC Supply Engineering Screen
 - 4.3 PAR Kicker Engineering Screen
 - 4.4 PAR Septum Engineering Screen
 - 4.5 Booster PS Quadrant Interlock Screen
- 5.0 Rampload
- 6.0 Composite Knobs
 - 6.1 Knobconfig
- 7.0 Magnet Conditioning
 - 7.1 Standardization
 - 7.2 Degaussing

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

MASTER

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED 85

1.0 Introduction

The purpose of this document is to provide comprehensive coverage of the APS power supply control design. This includes application software, embedded controller software, networks, and hardware. The basic components will be introduced first, followed by the requirements driving the overall design. Subsequent sections will address each component of the design one by one. Latter sections will address specific applications.

1.1 Components

The physical layout of power supply control follows the general APS controls model.

Fig. 1. Power Supply Control Layout

The workstations and Input/Output Controllers (IOC) are a shared resource. The EPICS control system provides the software/hardware infrastructure necessary to implement the control protocol between the workstations and IOC's. A bitbus subnet, specific to power supply control, is dropped off from various IOC's. Each subnet controls roughly 6 Power Supply Control Units (PSCU). Each PSCU can control up to 8 individual power supplies.

1.1.1 Power Supply Control Unit (PSCU)

The PSCU consists of Gespac, Inc. G-64 bus hardware running the OS-9 operating system. The typical unit has 21 3U form factor slots. Multi-tasking embedded control software runs on a M68000 or M68030 CPU. The software services external requests from both a local serial port, and the bitbus link. A variety of commercial and custom boards are present on the bus which control the remote DAC/ADC board in the actual power supply.

1.1.2 IOC

The IOC is a 21 slot VME crate with a Motorola 167 CPU running the VxWorks operating system. These are distributed about the facility and are shared (RF, PS, Vac, Diags.). A collection of EPICS records are configured for each attached power supply. Readbacks are obtained from the PSCU via periodically scanned records, and setpoints are delivered on demand. A variety of links are established between records to implement more complex functions such as tolerance checking.

1.1.3 Workstation

The typical workstation is a Sun SparcStation running SunOS4.1.x. The windowing system is SAIC_VUE running on an X11R5 server. Monitoring and control points, which are distributed among the many IOC's, are accessed via the Channel Access (CA) library. EPICS contains many high level tools which utilize CA. Custom applications may also be constructed using the CA library. Specific power supply applications are addressed later in this document. Typically, one needs to provide status monitoring, lattice

save/restore, ramp table loading, and setpoint knobbing.

1.2 Control Requirements

Control of power supplies can be classified into three major types, DC, ramped, and pulsed. The control hardware and software is different for each type, but is uniform within a type. In addition to supporting these three general types of control, a number of specific requirements are explained below which have significant implications for the design.

1.2.1 DC

The PAR, transfer line, and Storage Ring supplies are all DC. DC implies that a single setpoint may be given, and the current will ramp linearly to that setpoint. The DACs which provide the reference voltage are pulsed up or down, one count at a time (1). Therefore, the ramp slope is selected by a combination of input clock frequency, and a programmable clock divider. The solid core PAR magnet supplies in particular must be clocked up/down at a precise frequency. For other magnets, the exact rate is less critical, something less than the slew rate of the supply being adequate.

1.2.2 Ramped

The booster dipoles, quads, sextupoles, and correctors are all ramped. The PSCU for these supplies utilizes an Arbitrary Function Generator (AFG) board. The AFG board contains a deep FIFO, each byte indicating whether the DAC should be pulsed up, pulsed down, or left alone. An appropriate sequence of values in the FIFO will result in the desired reference voltage waveform being generated as the bytes are clocked out. Software is provided for describing a function with a particular number of segments. The remaining points required to fill the FIFO are generated via linear interpolation.

Under normal 2Hz booster operation, the ramping waveform is generated over a 250 ms period. The remaining 250 ms allow the AFG board to return the power supply to a DC bias level. A new waveform may be loaded at any time, and upon being selected, will apply to the next ramp cycle. A beam coast mode prevents the DAC from being reset after the first 250 ms. Once in beam coast mode, the current DAC value may be adjusted as though it were a DC supply.

1.2.3 Pulsed

The kicker, bumper, pinger, and septum magnet supplies are of the pulsed power supply class. These supplies are unique in that their operation is controlled primarily via pulses from the fast timing and event systems. The DACs are synchronously loaded with setpoints. The supply acts on the setpoints when given the appropriate sequence of level, charge, and discharge pulses.

1.2.4 Hardware Synchronized Sets

In addition to setting power supplies individually, it is necessary to command a set of power supplies to act on a new setpoint or ramp table simultaneously. In the synchrotron, for example, new waveforms may be prepared for the quadrupoles and correctors. The AFG boards for these supplies must simultaneously switch from the old waveforms to the new ones. The kickers may also be pre-loaded with new setpoints which will be acted on simultaneously.

1.2.5 Knobs

The ability to knob the setpoint(s) of DC supplies must be provided. These knobs are "relative" in that rotation causes the setpoint to move up or down from the last assigned point. A knob may control a single supply. A knob may also control a set of supplies by using a weight vector to specify the proportion of change for each supply. Typically the knob generates a stream of delta values. Each of these results in a

new setpoint for the supply. When knobbing a set of supplies, one must ensure that all the supplies in the set have responded before issuing another change.

2.0 Power Supply Control Units

The following figure shows the typical PSCU in relation to a single power supply. Keep in mind, however, that up to 8 power supplies may be interfaced.

Fig. 2. Signals between PSCU and a typical power supply.

2.1 Introduction

Each PSCU has one CPU. For DC supplies, a 68000 based CPU is used (GESSBS-6AH256). For the ramped supplies, a 68030 based CPU is used (GESMPU302) to provide hardware floating point capability for interpolating ramp tables. The CPU provides two serial ports. One is used for local dumb terminal control of the PSCU (for the 68030 CPU, the serial ports are on a companion board GESMFI-1). The control software is developed on a disk-based system, and burned onto EPROMS.

Either a timer card (DC supplies), or an AFG card (ramped supplies) is used to control the power supply DAC. All DAC's are pulsed up and down, as opposed to being parallel loaded. The one exception is when the supply is being used in the orbit control feedback system, in which case an absolute setpoint is loaded directly into the DAC.

The ADC-DAC interface card is used to control the DAC control source (either PSCU has control, or diagnostic feedback system does, not both). In addition, this interface card is used to acquire the high resolution ADC value from the current transducer. The actual A/D conversion occurs in the Remote DAC-ADC board. The data is then sent serially to the PSCU. The user also has the option of acquiring the actual DAC contents as well, hence the need for a Data Src select.

Lastly, commercial 12bit ADC and Digital I/O cards are used to interface with assorted field signals. Typically the 12bit ADC converts the magnet temperature, IGBT temp, capacitor temp, and PS voltage. The Digital I/O trips and resets the supply and monitors various interlock states.

2.2 Development

The embedded control software was co-developed by Oscar Despe and Claude Saunders. GESPEC,

Inc. provides the development platform. This platform is identical to the target stand-alone system with the exception that a floppy and hard disk are present with the editors and compilers needed. The operating system used is Microware's OS-9/68k V2.4. Code is edited and compiled on the development system. Testing is performed by serial-downloading the OS-9 modules to a stand-alone PCSU with special development EPROMS. After testing, S Records are generated and (somehow) transferred to an MS-DOS format disk. These S Records may then be used with any compatible EPROM burning system.

The complete development setup is shown below. The IOC/workstation component will be explained in a later section.

Fig. 3. PCSU development configuration.

A terminal session on the workstation is opened with the /term port of the Gaspac development system. The /term device is the serial port which corresponds to stdin and stdout in the OS-9 C environment. The PCSU code is compiled into a collection of OS-9 modules. Each of these modules is downloaded to the standalone PCSU via the /t1 serial port. Local control of the standalone system then occurs via the /ttyb -> /term -> /t1 -> /term path. By default, the PCSU software boots in remote (bitbus) control mode. Before local serial port control may be used, the PCSU must be switched to local control by entering the command "local".

While development and testing may both be done on the development system, this is not recommended. Mistakes in addressing boards may result in corrupting the hard disk.

In a similar fashion, the IOC code is developed on the workstation and loaded to the IOC via ethernet. Local control of the IOC is achieved via the serial port.

2.3 Code Overview

The PCSU software consists of three tasks: tpacu, rs232, and bitbus. Each task is a separate OS-9 module. In addition to these custom modules, there are a number of OS-9 drivers and OS support code that is part of any standalone system. We begin by describing the purpose of each C source and header file. An overview of how these modules work together during operation is then provided.

- main.c Tpacu (tasked power supply control unit) task. Shared memory (data module) initialization. Hardware initialization. Task forking and management.
- shmem.h Definition of shared memory structure. Shared memory is used to permit both the bitbus task and the rs232 task to share common data.
- my_defs.h Definition of board base addresses and assorted configuration parameters.
- rs232.c Rs232 task. Manages serial port interaction. Parses local control commands and executes them.

- timer.c Functions for operation of the custom timer card.
- pscu_9513.h Definition of AMD9513 timer chip registers.
- shfunc.c Functions for operation of ADC–DAC interface card and commercial boards (12bit ADC, Digital IO).
- afg.c Functions for operation of the AFG (Arbitrary Function Generator) board.
- bitbus.c Bitbus task. Initializes bitbus board and software.
- fsm.c Processes incoming bitbus messages from IOC and invokes appropriate functions.
- psMsg.h Definition of bitbus message commands.
- bblib.c Functions for dealing with GESFIL bitbus board (and driver software).
- bb.h Definition of bitbus message structure and bitbus protocol codes.

The tpscu task is always run first. This task begins by creating a shared data module and initializing most of the hardware. The rs232 and bitbus tasks are then forked. Tpsc then waits until the rs232 task exits before terminating itself. Should either the rs232 or bitbus task terminate abnormally, the exit status will be captured and printed.

The rs232 task begins by connecting to the shared data module. An intercept routine is then installed to handle incoming characters. The task sleeps until the intercept routine sees a newline character, at which time the task is woken by a signal and proceeds to parse and execute the command. It is necessary to intercept each character (unbuffered stdin) due to the half–duplex nature of the serial driver. If buffered stdin were used, no other task may write to stdout while the rs232 task is waiting for a new command line. This is unacceptable since the bitbus and tpscu task may write to stdout at any time.

The bitbus task begins by connecting to the shared data module. The /gesbit device is then opened. A loop is entered and the task immediately sleeps in the bitbus driver message receive call. Each new message arriving from the IOC wakes the task up, at which time the message is parsed and executed. A response message is always returned after which the task returns to sleep.

Commands, whether they are received from bitbus or the serial port, modify the shared data memory. This memory typically contains the readbacks and setpoints for each of the power supplies, as well as some general data for operating the hardware. There is no data synchronization problem between the rs232 and bitbus tasks since the PSCU is either in local or remote mode. When in remote (bitbus) mode, the serial port will accept no command except the one which selects the local mode. When switched to local mode, the bitbus task will not process "set" commands, and will only read from the current memory in response to "get" commands (as opposed to actually performing an A/D conversion, for example).

2.4 Generating EPROMs

Three separate EPROM versions are generated from the same source code using conditional compilation and makefile directives. One version runs on the 68000 based PSCU's and is for DC and pulsed supplies only. The second version runs on the 68030 based PSCU's and is for the 12–bit booster correction supplies. The third version also runs on the 68030 and is for the 16–bit booster supplies. The latter two versions are necessary to handle the two different AFG board designs.

The source code directory contains three subdirectories, 68K, 68020–12, and 68020–16. 68020 is used, since the compiler actually generates 68020 code, although it is run on an 030 processor. A directive at the beginning of the makefile indicates where the executables will be placed after compilation.

In addition to makefile directives, a number of #define's should be modified in the file my_defs.h according to the version being generated. These are documented at the head of the file: AFG to indicate whether an AFG board is present in the PSCU, NCH to indicate the number of channels supported (usually 4 or 8), FDEPTH to indicate the afg FIFO depth, IOADDR to indicate the base of synchronous address space on

the bus, and lastly VER to provide the EPROM version number.

Once compiled, the three OS-9 modules tpscu, rs232, and bitbus, must be copied into the appropriate directory for generation of the EPROM S-records. The 68K modules must be copied into /h0/sbs6/cmds/bootobjs. The 68020-XX modules must be copied into /h0/mpu302/cmds/bootobjs. A shell script entitled make_rom is then run to generate the S-record files.

The S-records are entitled U15 and U16 for the 68K target, and U30, U32, U50, U52 for the 68030 target. These files must then be transferred to an MS-DOS format disk for use in the EPROM programmer. Options abound here, but typically kermit may be used to transfer them to a PC or workstation.

2.5 Local Serial Control

Assuming the PSCU software has been loaded to a target system (either by download or EPROMS), we now examine the command set available from the local serial port. After booting and initialization, the PSCU is by default in remote control mode (i.e. the bitbus link has control). The PSCU must first be switched to local mode by entering the command "local" in lower case. Note that this will disable remote control, so it should only be done after informing the control operator. When the local session is complete, either reboot, or explicitly enter "remote" to return to the default mode. The available commands are as follows (note that the afg commands are disabled on the DC PSCU versions):

- local Switch to local serial port control of the PSCU
- remote Switch to remote (bitbus) control of the PSCU
- rampto ps int Ramp the DAC of power supply ps to absolute value int. int is a decimal number from 0 to 65535. This value will be interpreted according to the DAC in the power supply. For example, the bipolar supplies are offset binary, not 2's complement. If ps is -n, you must supply n int values. Supplies 0 to n will ramp up to their respective setpoints.
- setbyte ps hexval Set the digital output byte of ps to hexval.
- setsync ps mode Set the power supply's sync mode where mode=0 (default) means no sync, and mode = 1 means enter sync mode. Subsequent rampto commands will not cause actual ramping until either a hardware sync pulse is received by the timer card, or the sync mode is disabled.
- setsource ps mode Set the control source for ps. Mode=0 (default) means PSCU controls the power supply's DAC. Mode=1 means the diagnostic feedback system controls the DAC via the serial data link on the remote DAC-ADC card.
- calibrate ps Execute a 1 second calibrate cycle on the precision ADC for ps.
- getdac ps Get the current DAC value for ps. The value printed is decimal. Note that the value is raw. For some DACs, the lower two bits are meaningless.
- getrb ps rb Get analog readback(s) for ps. This command causes actual conversions to take place. Rb 0 is the precision ADC readback, while rb 1 through 4 are the four 12-bit GESADC readbacks. If rb=-1, all five will be reported. If ps=-1, the readbacks for all 8 power supplies will be reported.
- getsb ps Get the digital input byte for ps.
- exit or quit Exits the PSCU application. Enter tpscu at shell prompt to restart.
- setverbosity level Sets the degree of debugging print statements. Level 0 means no output will be generated other than to return data. Level 1 and up means some diagnostic information will be printed, particularly for the rampto command.
- setformat type Sets format to decimal or hex for data returned by "get" commands. Type 0 for hex output, type 1 for decimal output.
- setadcdelay delay Adjustable delay for operating the fast precision adc. Default of 10 seems to work well, but we provide this option as a precaution. The delay is actually a loop count, not a real time value.
- enabledacrb ps Use actual DAC readback as current value when determining how far to ramp up or down for a rampto command. This is default.

- disabledacrb ps initval Use DAC value maintained in software as current value, not actual DAC value. This capability is mainly for testing purposes, since the hardware DAC readback may occasionally fail, making other tests difficult. An initval must be provided to initialize the current value. Initval is a decimal value from 0 to 65535.
- ver Prints current EPROM version (V1.2.0 for example).
- dowaveform ps from to rate iter sync Generate triangle waveform that spans from to using divider value of rate. Do it iter times where -1 means indefinitely. If sync=1, wait for sync pulse to start. Note that this prevents any other use of the PSCU. It is only for testing purposes.
- setswcg ps freq bias peak imax Load SWCG fifo with data based on parameters given. Freq is between 2.0 and 125.0, bias and peak are in amps, and imax is the full scale current for the given ps type.
- setsample ps delay Configure the sample trigger on the SWCG board to read the precision ADC delay microseconds after the start pulse. Subsequent getrb ps 0 will fetch this value. Note that this is a one shot configuration. It must be repeated for each sample retrieved.
- setaf ps npts <<t1> <s1> ... <tn> <sn>> Load background afg fifo with new ramp table (up to 20 pts). If npts=-1, load it with the last known table.
- setafgmode ps mode mode=0 auto, mode=1 manual, mode=2 initialize
- supd ps Include the power supply in the set for next fifo swap.
- iupd ps Swap background and foreground fifos on next ramp cycle.
- setbias ps int Program reset bias for afg. Applied on next fifo swap.
- setkicker ps lid level Set kicker level in preparation for loading to DAC. lid=0 for injection level, lid=1 for extraction level.
- loadkicker Shift new kicker levels to DAC on next charge pulse.

2.6 Remote Bitbus Control

This section covers the format of bitbus messages sent to the PSCU's bitbus controller, and the responses returned. They correspond roughly to the serial commands, although much fewer in number. These bitbus messages are packaged and sent by the various device support modules in the IOC. Messages originate only from the master bitbus node on the IOC. However, when a message is received by the PSCU, a response must always be sent back.

The bitbus message structure is as follows. The fields are standard and documented elsewhere. We will focus only on the cmd_resp, sd_tasks, and data fields, for these contain application specific information.

```
typedef struct BITBUS_msg {
    u_short link;
    u_char length;
    u_char flags;
    u_char node_addr;
    u_char sd_tasks;
    u_char cmd_resp;
    u_char data[248];
}
```

We have chosen to use the lower nibble of the sd_tasks field (normally for encoding the destination 8044 task) to encode the power supply number to which the message applies. The cmd_resp field contains the command code for that message. All command codes begin at 0xC8, the user definable range as per BITBUS specifications. The file psMsg.h identifies the command codes placed in this field.

All parameters for a given command code are in the data field. We will examine the format of the data field for the command and the response. No field, other than the data field, need be modified by the PSCU application when formulating a response.

- GET_READBACKS (0xC8) – get adc readbacks

data byte	Parameters	Response
0	signal #	byte 1
1		byte 2
2		byte 3
3		

The signal # ranges from 0 to 5. Signal 0 requests the precision ADC readback which is returned in bytes 1 to 3, LSB first. Signal 1 to 4 requests one of the four 12bit ADC readbacks, which is returned in bytes 1 to 2, MSB first. Signal 5 requests the DAC readback which is returned in bytes 1 to 2, MSB first.

- SET_CURRENT (0xC9) – ramp ps to given DAC value

data byte	Parameters	Response
0	byte 1 (MSB)	ack code
1	byte 2	
2	byte 3	
3	byte 4 (LSB)	

The desired DAC value is supplied in bytes 1 to 4. Since DACs are mostly 16 bit, bytes 1 and 2 are usually 0. An ack code of 0x00 indicates all is well. No error return code system is currently implemented.

- RESET (0xCA) – load a new afg ramp table

data byte	Parameters	Response
0	sequence #	ack code
1	setpt/time data	
2	setpt/time data	
3	...	

A ramp table is sent using 12 bitbus messages. The sequence # identifies which. The data supplied in bytes 1 to 12 varies with the sequence #. Sequence #0 to #3 are used to send 20 setpoints as unsigned shorts. Sequence #4 to #11 are used to send 20 time values as unsigned ints. This forms the 20 entry time/setpoint table for the PSCU.

- GET_PSSB (0xCB) – get power supply status bit (from digital input byte)

data byte	Parameters	Response
0	bit # 0-7	value (0 or 1)
1		
2		
3		

The first parameter selects which bit of the digital input byte to read. The value (0 or 1) is returned.

- SET_RRAT (0xCC) – set ramp rate divider value

data byte	Parameters	Response
0	byte 1 (MSB)	ack code
1	byte 2	
2	byte 3	
3	byte 4 (LSB)	

The desired ramp rate clock divider value is supplied in bytes 1 to 4, MSB first. Value may range from 2 to 65535.

- CAL_PADC (0xCD) – execute calibration cycle on precision ADC

data byte	Parameters	Response
0	none	ack code
1		
2		
3		

- SET_PSCB (0xCE) – set power supply control bit (in digital output byte)

data byte	Parameters	Response
0	bit # 0-7	ack code
1	value (0 or 1)	
2		
3		

The first parameter indicates which output bit is to be set. The second parameter is the binary value to write.

- SET_SYNC (0xCF) – set power supply sync mode

data byte	Parameters	Response
0	mode (0 or 1)	ack code
1		
2		
3		

Mode 0 indicates no sync mode. Mode 1 indicates enter sync mode.

- SET_CNTS (0xD0) – set power supply control source

data byte	Parameters	Response
0	source (0 or 1)	ack code
1		
2		
3		

Source 0 sets the DAC control source to the PSCU (timer card). Source 1 sets the DAC control source to the external diagnostic input.

- GET_LOCAL (0xD1) – get PSCU local/remote mode (ps # does not apply for this command).

data byte	Parameters	Response
0	none	local
1		
2		
3		

Local byte is 1 if PSCU is in local mode, 0 if in remote mode (default).

- SET_BIAS (0xD2) – set AFG reset bias level (or relative DC adjust in manual mode)

data byte	Parameters	Response
0	byte 1 (MSB)	ack code
1	byte 2 (LSB)	
2		
3		

Programs AFG counter to generate given reset bias on next background/foreground swap. Bias level follows offset binary encoding for bipolar supplies. This same command will adjust the DC setting of the supply when in manual mode.

- SWAP_FIFO(0xD3) – set AFG control mode

data byte	Parameters	Response
0	mode	ack code
1		
2		
3		

Sets AFG operational mode. Mode 0 sets AFG to auto mode (normal 2Hz operation). Mode 1 sets AFG to manual mode (clock out one ramp and hold, allowing for DC adjustments with SET_BIAS). Mode 2 is an AFG reset mode, which should be entered prior to returning to auto mode. When the AFG is in auto mode, mode 3 requests a fifo foreground/background swap on the next ramp cycle. When the AFG is in

manual mode, mode 3 requests that the latest SET_BIAS level be pulsed out (DC adjust).

- SETKLEV (0xD4) – load PSCU kicker board with new kicker level

data byte	Parameters	Response
0	byte 1 (MSB)	ack code
1	byte 2	
2	byte 3	
3	byte 4 (LSB)	
4	level (0 or 1)	

The desired kicker setpoint is supplied in bytes 1 to 4. Since DACs are mostly 16 bit, bytes 1 and 2 are usually 0. The last byte should be 0 if supplying the injection level and 1 for extraction level. An ack code of 0x00 indicates all is well.

- LOADKLEV (0xD5) – arm new kicker levels to be shifted to remote DAC on next charge pulse

data byte	Parameters	Response
0	none	ack code
1		
2		
3		

Programs AFG to swap foreground with background on next ramp cycle. Mode tentatively used to distinguish between SUPD, IUPD, and MANUAL–IUPD type swaps.

3.0 IOC

The IOC's contain a database of EPICS records (process variables) required for power supply control. This section documents the records used for control of the various power supply types. The IOC hardware for control consists primarily of a PEP Modular Bitbus Controller.

3.1 Process Variable Nomenclature

All process variables which are involved in controlling a particular supply are prefixed with the physicists nomenclature. This is followed by a colon, a word describing the records purpose, and finally suffixed with the record type in capital letters. In short:

P1H1:CurrentAO	Par horizontal corrector, Current setpoint, Analog Output record
PQ1:DacAI	Par quadrupole, DAC readback, Analog Input record

3.2 Connecting a Database Record to Power Supply Hardware

A database record is associated with a particular channel of a power supply control unit via the INP/OUT and DTYP fields of that record. This section covers the format of the INP/OUT field and the available DTYP (device type) field values.

The INP/OUT field specifies which bitbus link, which power supply control unit, and which channel of that power supply control unit the record is to be associated with. Occasionally a parameter value is used in

conjunction with the DTYP field to specify, for example, which digital input bit of that supply you are interested in. The INP/OUT field appears as follows:

#L0 N1 P0 S3 @ L is link, N is node, P is power supply, and S is parameter

This entry points to bitbus link 0, node 1 (each power supply control unit is one node), and power supply 0 of that unit. If the DTYP field was "PSCU Status Byte", we would also be requesting S3, or the fourth digital input bit for supply 0.

The DTYP field selects a particular function of the power supply control unit. The supported functions and associated parameter (S) values are as follows:

PSCU ADC	12, 16, and 18 bit ADC readbacks
S=0	raw 18 bit precision adc readback
S=1	raw 12 bit gespac adc readback #1
S=2	raw 12 bit gespac adc readback #2
S=3	raw 12 bit gespac adc readback #3
S=4	raw 12 bit gespac adc readback #4
S=5	raw 16 bit dac readback
S=6	same as S=0, but shifted two bits to the right
S=7	same as S=6, but the 2's complement is taken
S=8	same as S=5, but the 1's complement is taken
PSCU Setpoint	16 bit DAC setpoint
18 PSCU Setpoint	18 bit DAC setpoint
PSCU Ramp Rate	Ramp rate divider setpoint
PSCU 12bit AFG bias	Arbitrary function generator reset level
PSCU 16bit AFG bias	
PSCU inj kick level	Kicker injection setpoint
PSCU ext kick level	Kicker extraction setpoint
PSCU Status Byte	Digital input bits (interlocks, status, etc)
S=0	status bit #1
...	
S=7	status bit #8
PSCU Local Mode	Control unit remote/local mode readback
PSCU Control Byte	Digital output bits (on, off, interlock reset, etc)
S=0	control bit #1
...	
S=7	control bit #8
PSCU PADC Calibration	Precision ADC calibration
PSCU Sync Mode	Timer card hardware sync mode
PSCU Control Source	DAC control source (timer card or feedback system)
PSCU AFG control	Arbitrary function generator operational mode
PSCU 12bit Ramp Table	Ramp table for 12 bit AFG board
PSCU 16bit Ramp Table	Ramp table for 16 bit AFG board
PSCU kick level load	Kicker level pre-load

3.3 PAR/LTP/PTB Quad, Sext, Corr, Dipole Trim Records

P1H1:CurrentAO	Current setpoint. (DESC field of record gives magnet name)
P1H1:CurrentCALC	Calibration Slope and Offset for CurrentAO
P1H1:Current2AO	Current setpoint which goes to actual PSCU.
P1H1:CurrentAI	Precision ADC readback (16 bits) (current transductor).
P1H1:DacAI	DAC readback.
P1H1:OutCurrentAI	12 bit output current readback.
P1H1:OutVoltageAI	12 bit output voltage readback
P1H1:DACOutVoltageAI	12 bit DAC output voltage readback.
P1H1:MagTempAI	12 bit magnet temperature.

PlH1:RateDividerAO Timer card ramp rate divider value.
 PlH1:StandardizeSUB Record used for standardization or degaussing.
 PlH1:SyncModeBO Sync mode bit on timer card.
 PlH1:StatusCALC Overall power supply status (interlocks, communications)
 PlH1:ResetSEQ Reset interlocks and turn power supply on.
 PlH1:Bit0BI Digital input 1 (DESC field of record describes purpose)
 ...
 PlH1:Bit7BI Digital input 8
 PlH1:CalibrateBO Precision ADC calibration cycle
 PlH1:OnBO not used directly (see ResetSEQ)
 PlH1:OffBO Supply off bit.
 PlH1:ResetBO not used directly (see ResetSEQ)

3.4 PAR Kicker Records

PlIK:VoltageAO Kicker injection setpoint (applied when ShiftLevels and charge pu
 PlIK:PFNVoltageAI Precision ADC readback (16 bits) (pulse forming net voltage)
 PlIK:OilTankTempAI 12 bit oil tank temperature readback
 PlIK:RackTempAI 12 bit power supply cabinet temperature
 PlIK:ThyFilVoltageAI 12 bit thyatron filament voltage
 PlIK:ThyResCurrentAI 12 bit thyatron reservoir current
 PlIK:DacAI DAC readback (may be injection or extraction level)
 PlIK:Bit0BI Digital input 1 (DESC field of record describes purpose)
 ...
 PlIK:Bit15BI Digital input 16
 PlIK:CalibrateBO Precision ADC calibration cycle
 PlIK:ShiftLevelsBO Prepare latest VoltageAO setpoints for transfer to the remote
 PlIK:OnBO Turn on/off high voltage
 PlIK:ResetBO Reset interlocks
 PlIK:StatusCALC Overall power supply status (interlocks, communications)
 PlEK:VoltageAO Kicker extraction setpoint (all other records shared with injectic

3.5 PAR Septum Records

PISP:CurrentAO Septum injection setpoint (applied when ShiftLevels and charge)
 PISP:CurrentAI Precision ADC readback
 PISP:DacAI DAC readback (may be injection or extraction level)
 PISP:IchAI 12 bit ICh readback
 PISP:ECfAI 12 bit ECf readback
 PISP:EC2AI 12 bit EC2 readback
 PISP:MagTempAI 12 bit magnet temperature
 PISP:Bit0BI Digital input bit 1 (DESC field of record describes purpose)
 ...
 PISP:Bit7BI Digital input bit 8
 PISP:CalibrateBO Precision ADC calibration cycle
 PISP:ShiftLevelsBO Prepare latest CurrentAO setpoints for transfer to the remote
 PISP:OnBO not used directly (see ResetSEQ)
 PISP:OffBO turn off supply
 PISP:ResetBO not used directly (see ResetSEQ)
 PISP:StatusCALC Overall power supply status (interlocks, communications)
 PISP:ResetSEQ Reset interlocks and turn on supply
 PESp:CurrentAO Septum extraction setpoint (all other records shared with injectic

3.6 Booster Temporary Quad, Sext, Dipole Records

B:QD:CurrentAO Current setpoint. (DESC field of record gives magnet name)
 B:QD:CurrentCALC Calibration Slope and Offset for CurrentAO

B:QD:Current2AO Current setpoint which goes to actual PSCU.
 B:QD:CurrentAI Precision ADC readback (16 bits) (current transducer).
 B:QD:DacAI DAC readback.
 B:QD:DCCurrentAI 12 bit output current readback.
 B:QD:DCVoltageAI 12 bit output voltage readback
 B:QD:ACCurrentAI 12 bit AC current.
 B:QD:ACVoltageAI 12 bit AC voltage.
 B:QD:RateDividerAO Timer card ramp rate divider value.
 B:QD:StandardizeSUB Record used for standardization or degaussing.
 B:QD:SyncModeBO Sync mode bit on timer card.
 B:QD:StatusCALC Overall power supply status (interlocks, communications)
 B:QD:ResetSEQ Reset interlocks and turn power supply on.
 B:QD:Bit0BI Digital input 1 (DESC field of record describes purpose)
 ...
 B:QD:Bit7BI Digital input 8
 B:QD:CalibrateBO Precision ADC calibration cycle
 B:QD:OnBO not used directly (see ResetSEQ)
 B:QD:OffBO Supply off bit.
 B:QD:ResetBO not used directly (see ResetSEQ)

3.7 Booster Temporary Corrector Records

B1C1V:CurrentAO Current setpoint. (DESC field of record gives magnet name)
 B1C1V:CurrentCALC Calibration Slope and Offset for CurrentAO
 B1C1V:Current2AO Current setpoint which goes to actual PSCU.
 B1C1V:CurrentAI Precision ADC readback (16 bits) (current transducer).
 B1C1V:DacAI DAC readback.
 B1C1V:IGBT1TempAI 12 bit IGBT1 Temperature.
 B1C1V:IGBT2TempAI 12 bit IGBT2 Temperature.
 B1C1V:CapTempAI 12 bit capacitor temperature.
 B1C1V:MagTempAI 12 bit magnet temperature.
 B1C1V:RateDividerAO Timer card ramp rate divider value.
 B1C1V:StandardizeSUB Record used for standardization or degaussing.
 B1C1V:SyncModeBO Sync mode bit on timer card.
 B1C1V:StatusCALC Overall power supply status (interlocks, communications)
 B1C1V:ResetSEQ Reset interlocks and turn power supply on.
 B1C1V:Bit0BI Digital input 1 (DESC field of record describes purpose)
 ...
 B1C1V:Bit7BI Digital input 8
 B1C1V:CalibrateBO Precision ADC calibration cycle
 B1C1V:OnBO not used directly (see ResetSEQ)
 B1C1V:OffBO Supply off bit.
 B1C1V:ResetBO not used directly (see ResetSEQ)

3.8 Booster Kicker Records

B:IK:VoltageAO Kicker injection setpoint (applied when ShiftLevels and charge pu
 B:IK:PFNVoltageAI Precision ADC readback (16 bits) (pulse forming net voltage)
 B:IK:OilTankTempAI 12 bit oil tank temperature readback
 B:IK:RackTempAI 12 bit power supply cabinet temperature
 B:IK:ThyFilVoltageAI 12 bit thyatron filament voltage
 B:IK:ThyResCurrentAI 12 bit thyatron reservoir current
 B:IK:DacAI DAC readback (may be injection or extraction level)
 B:IK:Bit0BI Digital input 1 (DESC field of record describes purpose)
 ...
 B:IK:Bit15BI Digital input 16
 B:IK:CalibrateBO Precision ADC calibration cycle
 B:IK:ShiftLevelsBO Prepare latest VoltageAO setpoints for transfer to the remote

B:IK:OnBO	Turn on/off high voltage
B:IK:ResetBO	Reset interlocks
B:IK:StatusCALC	Overall power supply status (interlocks, communications)

3.9 Booster Septum Records

B:ISP:CurrentAO	Septum injection setpoint (applied when ShiftLevels and charge)
B:ISP:CurrentAI	Precision ADC readback
B:ISP:DacAI	DAC readback (may be injection or extraction level)
B:ISP:IChAI	12 bit ICh readback
B:ISP:ECfAI	12 bit ECf readback
B:ISP:EC2AI	12 bit EC2 readback
B:ISP:MagTempAI	12 bit magnet temperature
B:ISP:Bit0BI	Digital input bit 1 (DESC field of record describes purpose)
...	
B:ISP:Bit7BI	Digital input bit 8
B:ISP:CalibrateBO	Precision ADC calibration cycle
B:ISP:ShiftLevelsBO	Prepare latest CurrentAO setpoints for transfer to the remote
B:ISP:OnBO	not used directly (see ResetSEQ)
B:ISP:OffBO	turn off supply
B:ISP:ResetBO	not used directly (see ResetSEQ)
B:ISP:StatusCALC	Overall power supply status (interlocks, communications)
B:ISP:ResetSEQ	Reset interlocks and turn on supply

4.0 Control Screens

The majority of operations carried out on power supplies involve non-power supply specific tools such as BURT (backup and restore tool), and km (knob manager). Screens are provided to check the status of all power supplies in a given ring or transfer line. Engineering screens are provided primarily for diagnosis. Should one of the status screens indicate a bad supply, the engineering screen may be brought up to investigate which interlock tripped. This section briefly describes the various control screens provided. For the moment, the temporary booster supplies and correctors have the same screens as the PAR, ie. DC operation only.

4.1 Summary Power Supply Status

Shown here is a status screen for the LTP. A similar screen is available for PAR, PTB, Booster, and BTS. This screen provides an led for each supply. A green led indicates the supply is available for use. Red indicates that the supply is off, an interlock has tripped, or that someone is operating the supply locally. White indicates that remote control of the supply is not possible due to failed communications. Either the IOC is down, or the bitbus link between the IOC and power supply control unit has failed. When white, the readbacks viewed on the engineering screen should be considered invalid. The engineering screen for each supply may be brought up by dragging the mouse down on the blue menu button.

4.2 DC Supply Engineering Screen

The name of the supply is always listed under Device in the upper left hand corner. From left to right are:

SET	DAC setpoint in the units given to the right.
READ	DAC readback. The DAC can be observed to transition from one setpoint to another
CURRENT	16 bit precision ADC readback of the current transducer.

UNITS Units for the SET and READ fields.

STATUS The summary status led is essentially the OR of all the individual bits just .
The colors follow an alarm priority from green to yellow to red to white. The summary led reflects the highest priority color from the bits to the right.

When the STATUS led is white, the data on the display should be considered invalid. A white led almost always indicates that communications with the power supply control unit have failed. A red Communications bit indicates the power supply is in local control. A green Communications bit indicates that the data on the screen is valid. The remaining bits vary from supply to supply, but generally reflect the interlock status. These bits will be either red or green.

The RESET/ON and OFF buttons provide the means to turn the supply off and on. The Calibrate button initiates a precision adc calibration cycle. This is generally done automatically on power up. The Rate Divider field allows adjustment of the DAC slew rate. The lower the divider, the higher the slew rate. This field will eventually be changed to reflect the actual slew rate.

4.3 PAR Kicker Engineering Screen

Note that this screen provides control over a shared kicker (injection/extraction). Each level has its own setpoint. For a non-shared kicker (ie. the booster kickers), there will be only one device listed.

This screen is essentially identical to the DC supply engineering screen with the exception of one button. When a new setpoint is given in the SET field, the new value is not sent to the DAC until the Send Level button is pressed. The new setpoint will go into effect on the next kicker charge pulse from the timing system. With this button, it is possible to prepare several kickers with new setpoints to be acted on simultaneously. The Send Level button need only be pressed once on one of the kicker screens.

The PFN Voltage readback (VOLTAGE) is sampled at a specific time relative to the Linac Pretrigger Gate Pulse. The time of sample may be adjusted via the blue text entry field under the VOLTAGE readback.

4.4 PAR Septum Engineering Screen

Note that this screen provides control over a shared septum (injection/extraction). Each level has its own setpoint. For a non-shared septum (ie. the booster septums), there will be only one device listed.

This screen is essentially identical to the DC supply engineering screen with the exception of one button. When a new setpoint is given in the SET field, the new value is not sent to the DAC until the Send Level button is pressed. The new setpoint will go into effect on the next septum charge pulse from the timing system. With this button, it is possible to prepare several septums with new setpoints to be acted on simultaneously. The Send Level button need only be pressed once on one of the septum screens.

The Primary Current readback (CURRENT) is sampled at a specific time relative to the Linac Pretrigger Gate Pulse. The time of sample may be adjusted via the blue text entry field under the CURRENT readback.

4.5 Booster PS Quadrant Interlock Screen

This screen monitors the grouped magnet interlocks. There is a chain of interlocks for each quadrant of magnets. They are also grouped by type: Dipole, Quadrupole Focusing (QF), Quadrupole Defocusing (QD), Sextupole Focusing (SF), Sextupole Defocusing (SD). A green led indicates no trip, red indicates a trip, and white means that communications with the IOC are down. The blue button will reset all the interlock chains..

5.0 Rampload

Support for loading ramp tables to ramped supplies is in the works. The tool will accept an SDDS specification of the supply ramp tables.

6.0 Composite Knobs

For knobbing of individual setpoints, see documentation on km (knob manager). A preliminary composite knob assignment tool called knobconfig is available. This tool reads a single SDDS file which specifies which knobs are to be assigned to which devices (or process variables). An overall gain may be given, and weights for the composite device constituents may also be supplied.

6.1 Knobconfig

NAME
knobconfig
SYNOPSIS
knobconfig spec-file
DESCRIPTION
Assigns knobs to devices and process variables according to spec-file. Assignments are active during program execution. In general, it is best not to run this utility in background. Knobs are activated in the order that data tables appear in the spec-file.
A typical spec-file is as follows:


```

-----
SDDS1
&description
    text="Knob Assignment Specification"
    contents="knob assignment specification"
&end
&parameter name=ControlName
    type=string
&end
&parameter name=ControlType
    type=string
&end
&parameter name=ControlUnits
    type=string
&end
&parameter name=Gain
    type=double
&end
&column name=ControlName
    type=string
&end
&column    name=Weight
    type=double
&end
&data    mode=ascii
    no_row_counts=1
&end
COMPDEVICENAME
dev
tune
0.01
P:Q1  1.0000
P:Q2  1.1134

PTB:Q3
dev
Amps
0.1

PTB:Q4:CurrentAO
pv
Amps
0.1

```

7.0 Magnet Conditioning

Two utilities are provided for conditioning the magnets for DC supplies. The utility standardize is for unipolar conditioning, while degauss is for bipolar conditioning. The following subsections describe these utilities in a manual page format.

7.1 Standardization

NAME

standardize

SYNOPSIS

standardize [-query|-configure|-stop] spec-file

DESCRIPTION

Standardizes the set of devices specified in spec-file. The spec-file must be in SDDS format. Each row in the spec-file contains the device name, range over which to standardize, time over which to standardize, number of cycles, finishing setpoint, and whether to approach the finish from above or below. Simply enter "standardize spec-file" to begin standardization. Standardize uses linear transitions.

A typical spec-file is as follows:

```
-----
SDDS1
&description text="whatever"
  contents="standardization specification" &end
&column name=ControlName type=string &end
&column name=ControlType type=string &end
&column name=From type=double &end
&column name=To type=double &end
&column name=Minutes type=long &end
&column name=Seconds type=long &end
&column name=NumCycles type=long &end
&column name=Finish type=double &end
&column name=Approach type=string &end
&data mode=ascii no_row_counts=1 &end
PQ1 dev 0.0 180.0 0 30 1 7.0 above
PQ2 dev 0.0 180.0 1 23 2 9.3 below
-----
From      - low end of standardization cycle span
To        - upper end of standardization cycle span
Minutes   - time in minutes to transition (from..to)
Seconds   - time in seconds to transition (from..to)
NumCycles - number of cycles in standardization (0 or more)
            if (approach == below)
              0 means transition (now..to..from..finish)
              1 means transition (now..to..from..to..from..finish)
              etc..
            else if (approach == above)
              0 means transition (now..from..to..finish)
              1 means transition (now..from..to..from..to..finish)
              etc..
Finish     - end standardization at this setpoint
Approach-  "below" means approach finish from below
            "above" means approach finish from above
```

OPTIONS

-query

Find out which devices in spec-file are undergoing standardization.

-stop

Halt standardization of devices in spec-file.

-configure

Configure subroutine records for standardization; Do not initiate. Standardization may then be initiated later via a channel access put to the PROC field of the subroutine record, or via a device message "devSend P1H1 standardize".

7.2 Degaussing

NAME

degauss

SYNOPSIS

degauss [-query|-configure|-stop] spec-file

DESCRIPTION

Degausses the set of devices specified in spec-file. The spec-file must be in SDDS format. Each row in spec-file contains the device name, decay time, sin wave period, peak current, and number of decay times desired. Simply enter "degauss spec-file" to begin degaussing. Degaussing involves a time-dependent current: $I(t) = I_{\text{Max}} * \exp(-t/\text{DecayTime}) * \sin(2\pi * (t/\text{period}))$

A typical spec-file is as follows:

```
-----
SDDS1
&description text="whatever"
  contents="standardization specification" &end
&column name=ControlName type=string &end
&column name=ControlType type=string &end
&column name=DecayMinutes type=long &end
&column name=DecaySeconds type=long &end
&column name=PeriodSeconds type=long &end
&column name=MaxCurrent type=double &end
&column name=NumDecay type=long &end
&data mode=ascii no_row_counts=1 &end
PlH1 dev 0 30 10 19.0 3
PlH2 dev 1 30 28 19.0 2
-----
DecayMinutes    - minutes for one decay cycle
DecaySeconds    - seconds for one decay cycle
PeriodSeconds   - seconds for one sine wave period
MaxCurrent      - max current for degaussing function
NumDecay        - number of decay times
```

OPTIONS

```
-query
Find out which devices in spec-file are undergoing degaussing.
-stop
Halt degaussing of devices in spec-file.
-configure
Configure subroutine records for degaussing; Do not initiate.
Degaussing may then be initiated later via a channel access
put to the PROC field of the subroutine record, or via a device
message "devSend PlH1 degauss".
```

Footnotes

(1)Supplies involved in feedback orbit correction allow for parallel loading of the DAC.