SOFTWARE DESIGN DOCUMENT

for the

DETECTOR CONTROL ELECTRONICS (DCE) FLIGHT SOFTWARE

for the

COSMIC ORIGINS SPECTROGRAPH (COS)

Contract No NAS5-98043
CDRL No. DM-03

Prepared for:

Goddard Space Flight Center
Greenbelt, MD

Prepared by:

Experimental Astrophysics Group, Space Sciences Laboratory
University of California, Berkeley

COS-UCB-00x

October 8, 1999

Prepared By:

Mr. Daniel Blackman
FUV Detector Software Lead

Reviewed By:

Mr. Geoff Gaines
FUV Detector Systems Engineer

Dr. Derek Buzasi
FUV Detector Scientist

Mr. Grant Blue
BATC COS Software Manager

Approved By:

Dr. Kenneth Brownsberger, University of Colorado, Boulder
HST-COS Software Scientist

Date

BLANK PAGE

TABLE OF CONTENTS

TABLE OF FIGURES

1 SCOPE

1.1 IDENTIFICATION

This DCE Software Design Document (COS-UCB-00x) is the top-level specification describes the software design of the FUV Detector Control Electronics Flight Software (DCE FSW) Computer Software Configuration Item (CSCI) for the Cosmic Origins Spectrograph (COS) instrument. The developer of the FUV Detector Subsystem is the Experimental Astrophysics Group (EAG) at the Space Science Laboratory, University of California, Berkeley.

This document is prepared in accordance with the requirements of a subcontract. This document satisfies the subcontract requirements for Flight Software Documents, DCE FSW Computer Software Configuration Item (CSCI). It is written using an example document provided by BATC as a guide.

1.2 DOCUMENT OVERVIEW

This DCE Software Design Document describes how the requirements that have been allocated to the COS DCE FSW are to be satisfied by design. The organization of this document is outlined below in a description of the contents of each of the ensuing sections.

Section 1 defines the scope of this document and introduces the DCE FSW.
Section 2 lists documents that have been directly or indirectly referenced by this document.
Section 3 contains a glossary of document acronyms.
Section 4 presents a bottom-level hardware interface and software design for the DCE FSW.
Section 5 presents a top-level hardware interface and software design for the DCE FSW.
Section 6 describes the FUV Detector Control Electronics hardware and software environments.
Section 7 appends some notes.

1.3 COS FLIGHT SOFTWARE OVERVIEW

Within the FUV Detector Subsystem, the DCE FSW provides the FUV detector with the computational, interface, and memory resources necessary for performing overall FUV detector control tasks. Commands are accepted and processed by the DCE FSW. The DCE FSW outputs engineering telemetry data (housekeeping). Note that while the DCE FSW controls hardware that generates Science Data, the DCE FSW sees only counts and not location information for each photon event. Science Data does not pass through the DCE FSW, but is output directly in FUV detector hardware.

The DCE FSW is the brain of the FUV Detector Subsystem, with an internal interface to the TDC, HVPS, and LVPS, known collectively as the Electronics Interface. The FUV Detector Subsystem includes a number of temperature, voltage, current, and other sensors that are converted to a digital format and reported by the DCE FSW in regular housekeeping updates. The DCE FSW will implement a command interface for the MEB, permitting direct control of the FUV detector. The DCE FSW will also perform checking of some limits and other self-tests.

1.4 RELATIONSHIP TO OTHER DOCUMENTS

This COS DCE Software Design Document describes the implementation in support of functionality of the COS DCE FSW. Software requirements are detailed in the DCE FSW Requirements Document (COS-UCB-004). Requirements traceability will be documented in matrix format in the Software Test Plan. Many details of FUV detector operations are described in the ICD (COS-UCB-001) and the Software User's Guide. The organization of the Command and Housekeeping interface to the MEB is described in detail in the FUV detector ICD.

2 APPLICABLE DOCUMENTS

These documents provide additional details concerning the design and implementation of features required of the DCE FSW.

2.1 COS RELATED DOCUMENTATION

Interface Control Document, FUV Detector Subsystem, COS-UCB-001
Configuration Management Document, FUV Detector Subsystem, COS-UCB-003
DCE FSW Requirements Document, COS-UCB-004
FUV Detector Software Configuration Management Plan, COS-UCB-005
Appendix to Software User's Guide, BATC
FUV Detector Software Test Plan, COS-UCB-008
FUV Detector Software Test Procedures
COS Control Section Flight Software Requirements Document

2.2 DCE HARDWARE DOCUMENTATION

MCS 51 Microcontroller Family User's Manual, Order Number 272383-002, Intel, Feb '94

3 LIST OF ACRONYMS AND ABBREVIATIONS

Aux	Auxiliary Power
COS	Cosmic Origins Spectrograph
BATC	Ball Aerospace & Technologies Corp.
CRC	Cyclic Redundancy Check
CSC	Computer Software Component
CSCI	Computer Software Configuration Item
DCE	FUV Detector Control Electronics
EAG	Experimental Astrophysics Group, Space Sciences Laboratory, UC Berkeley
FSW	Flight software
FUSE	Far Ultraviolet Spectrographic Explorer
FUV	Far Ultraviolet
GSE	Ground Support Equipment
GSFC	Goddard Space Flight Center
HST	Hubble Space Telescope
HV	High Voltage
HVPS	High Voltage Power Supply
Hz	Hertz, cycles per second
ICD	Interface Control Document
ICE	In-Circuit Emulator
I/O	Input/Output
I&T	Integration & Test
LVPS	Low Voltage Power Supply
MEB	Main Electronics Box
NASA	National Aeronautics and Space Administration
PROM	Programmable Read-Only Memory
PHD	Pulse Height Distribution
RAM	Random Access Memory
ROM	Read-Only Memory
SEU	Single Event Upset
SRD	Software Requirements Document
TDC	Time-to-Digital Converter
UCB	University of California, Berkeley

4 PRELIMINARY DESIGN

4.1 DCE HARDWARE

From the perspective of the FSW, the DCE Hardware is memory and some i/o ports. An understanding of the DCE hardware layout is essential to the design of the FSW. The 8051 communicates with DCE hardware through both ports and memory mapped i/o.

The hardware diagram below shows the hardware components with their annotated memory-mapped i/o address, where applicable.

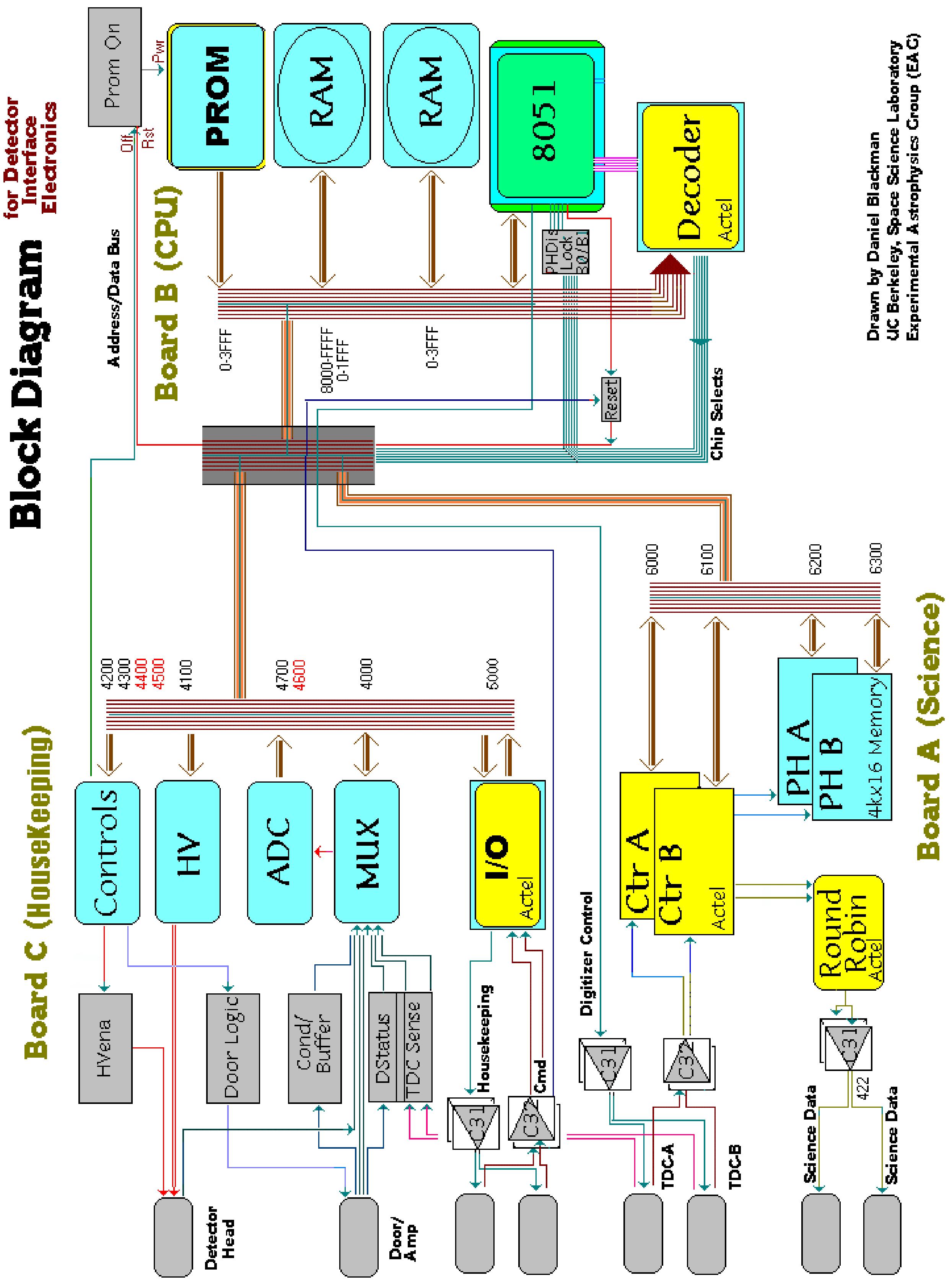

Figure 1: DCE Block Diagram

4.1.1 8051 MICROCONTROLLER

The DCE uses the UTMC68RH051 (8051), which is a close variant of the 8051FC microcontroller. Software for the 8051 is installed in the DCE as a PROM device, which the 8051 reads from address 0 in the event of any reset. The 8051 will execute code as long as it is powered (and clocked) and has no idle state. See a production specification and programming guide for the 8051FC.

4.1.1.1 Ports

Ports are control bits that the 8051 controls directly. The digitizer interface is a synchronous serial port that uses several of these control bits. The 8051 can select some memory mapping options. The 8051 will use a port bit to inhibit updates to PH-histogram data while reading it. Many of the requirements for the software use of port bits are driven by the hardware design.

4.1.1.2 Memory Mapped I/O

Most control of the DCE is supported by memory-mapped i/o. Resources are allocated specific addresses in memory. When written-to, a register in some other DCE hardware component – not memory – is updated. When read-from, a register supplies data from some DCE hardware component and not main memory.

Inputs: command traffic, switches from door & hv, temps/volts/currents, counters, phd, serial port, pca timer

Events: timer tick, door timeout, door latch, reset, ilimit shutdown, crp trigger

Functions: handle commands, check door, check hv, check counters, compute crcs,

Outputs: housekeeping, digitizer settings, door ctrl, hv ctrl, serial port, LEDs, reset pin

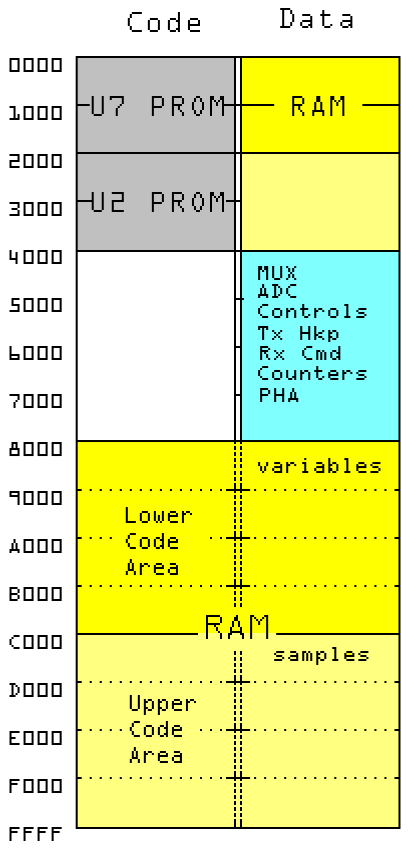

Figure 2: Memory Map

4.1.2 Hardware Components

8051 based computer stores executable code in PROM, stores results and other data in RAM, reads and writes hardware through Memory Mapped I/O, use of PROMs, intent to run operate state from RAM, need to upload code requires partial Von Neumann architecture.

4.1.2.1 PROM

The programmable read-only memory (PROM) device is programmed and installed in hardware. It contains code that initializes to the Boot state, which cannot be changed in flight. The PROM may also contain an operate state code image that can be copied to RAM and executed on command.

4.1.2.2 RAM

There are two random access memory (RAM) devices named 8K RAM (addresses 0..0x4000) and 32K RAM (addresses 0x8000..0xFFFF). As a design choice, memory is allocated to specific purposes somewhat arbitrarily: data passing to and from the CS (ie command/housekeeping traffic) has been located in 8K RAM; code space, program variables and private data and samples are all stored in certain locations in 32K RAM. The 8K RAM is completely initialized with zero values upon power-on reset only. The 32K RAM is never initialized uppon power-on reset.

4.1.2.3 Digitizer

The digitizer is the electronics that interpret photon events. Event criterion can be controlled on command, which are programmed via a synchronous serial link implemented with 8051 port pins. The digitizer returns setting talkbacks as an input to the MUX/ADC. The digitizer is programmed with default values upon power-on reset, and after that, settings are modified only on command.

4.1.2.4 HVPS DACs

The HVPS converts a 0-to-5V DAC setting to a HV value in the 2000-6000V range at the detector head. The exact conversion coefficients for the DAC-to-HV relation are specified in the list of Housekeeping items.

4.1.2.5 Control Registers

Control bits for the door and HV are latched in one of two control registers. The Door and HV control bits that could inflict some "severe degradation" upon the FUV detector have been allocated to one control register that has been equipped with Lock control bit. The Lock bit is described in section 5.1.1.5 Control Lock.

4.1.2.6 Counters

Photon events increment counters and PH bins that appear as memory-mapped output. There are several counters because they count events at different stages of digitizer processing, which throws out some photon events that do not meet some criterion specified by the digitizer and its threshold settings.

4.2 DCE DESIGN IN CONTEXT OF 8051

The 8051 was selected to control the DCE hardware because of its reliability and thermal qualities and simplicity suited to the functions allocated to the software.

4.2.1 EXTERNAL MEMORY: CODE AND DATA

The organization of software code and data in the 8051 is known as Harvard Architecture, whereby code and data each occupy a distinct 64K address space. The 8051 knows to access code memory when it fetches an instruction opcode, and to access data memory when it reads or writes a variable. The 8051 provides all information required to select the appropriate memory address at its pins. It is up to the DCE hardware designer to actually implement the Harvard memory design in hardware. In fact, the DCE memory layout above addresses 0x8000 are designed as Von Neumann Architecture, which code and data occupy the same logical memory space.

The arrangement allows code to start execution in PROM (code space), write code into RAM as data, and then execute that code out of RAM. See Figure 2, Memory Map.

Memory Regions are contiguous subsets of the 64K code and data spaced defined for the purpose of computing a characteristic CRC value for comparison against previous computations. This use of CRC computations to characterize a DCE software version allows operators to know whether a given software load is corrupted. An uncorrupted software image will have a predictable CRC value computed on the applicable code space region.

4.2.2 INTERNAL REGISTERS

The 8051 contains 256 (0x100) addressable registers. Above address 0x80, not all registers are implemented, and most of those directly control the 8051 itself. Such registers include timers, serial port data, interrupt control, and port bits.

4.3 SOFTWARE DEVELOPMENT

The DCE FSW is being developed from heritage FUSE software. While there are some important differences between FUSE and COS (and GALEX for that matter!), efforts have been made to keep the software similar, if not the same.

4.3.1 PROGRAMMING LANGUAGE

The DCE software is written in 8051 Assembly Language. There is no RTOS. All aspects of control flow and resource allocation are under the control of the software programmer.

4.3.2 SOFTWARE TOOLS

The drawing below depicts the process of building executable code from source code using the set of software tools.

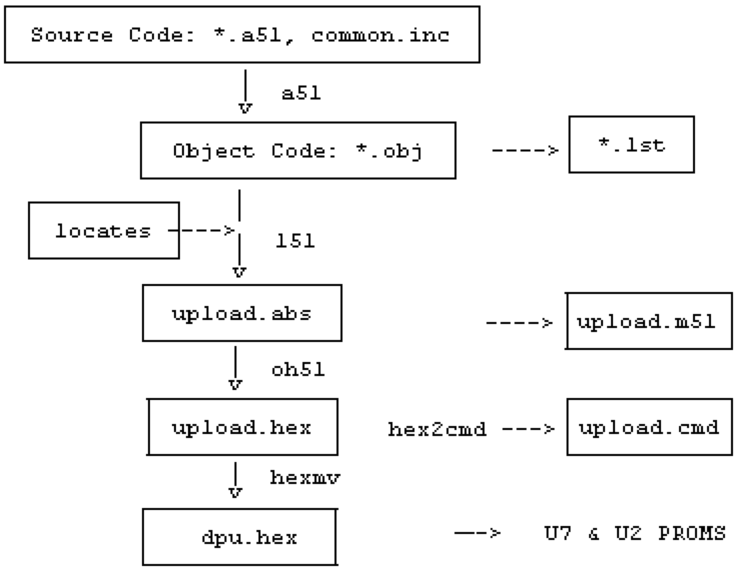

Figure 3: Software Build Flow

Required software tools include: the a51 assembler, l51 linker/locator, oh51 object-to-hex, hexmv relocator, hex2cmd scripting tool, text editor, 8051 emulator, PROM programmer, and an OS for the host computer.

Locates is the name of a file containing specific address directives to be used when linking the various software modules.

The output files are:
.LST the assembled listing. Useful for exactly locating any software entity (like addresses of variables, text of 8051 code, and contains a copy of the source code.
.M51 the locator map. Useful for identifying the starting address of a software module and an important part of location the address of software entities of interest.
.CMD an uploadable script in FUSE eGSE format.
.HEX a text code image used in the creation of PROMs.

4.4 DATA FLOW DESIGN

The DCE FSW is responsible for processing the flow of data between the FUV detector and COS/MEB/CS. The 8051 also produces timing data that drives some DCE tasks. Incoming command traffic drives the Command Handler Task. The Housekeeping Task waits for a signal to transmit the contents of Housekeeping Data, which is a region of memory reserved for engineering data made available to report the DCE's internal state.

Figure 4, Simplified Data Flow diagram shows how data flow through the DCE is driven by commanding from the CS, on one side, and photon event from the detector head on the other. Photon events change the state of the DCE only when excessive counts trigger the Count Rate Protection mechanism. The CDC/TDC is another notation meaning "digitizer." Similarly, the sensors (Temps/Volts/Current) can change the state of HV to off if excessive (over the set limit) current is sensed by the Current Limit Protection Task.

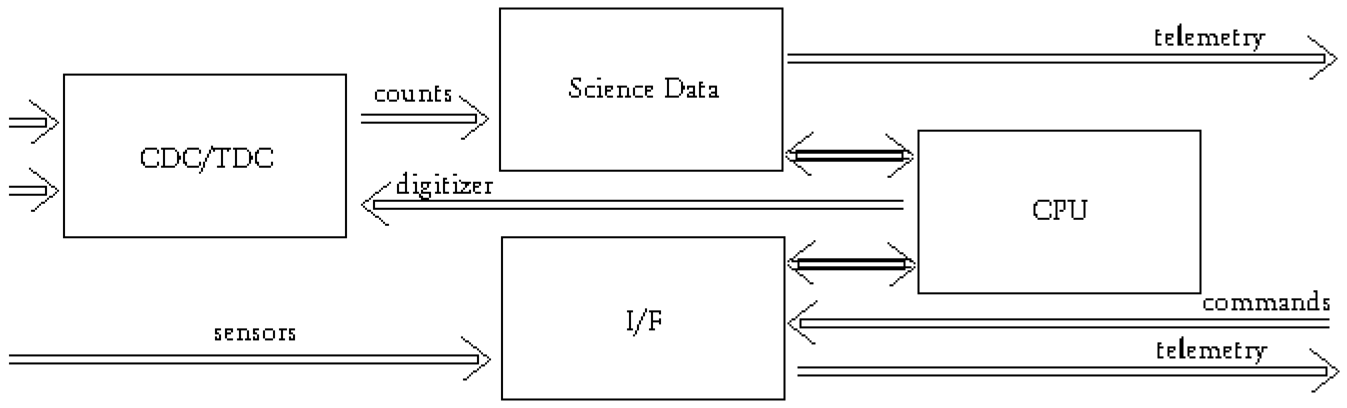

Figure 4: Simplified Data Flow Diagram

4.4.1 DATA FLOW INPUTS

4.4.1.1 Command Traffic

Incoming command traffic is stored in memory by the Command ISR. The Command Handler Task will recognise a new op-code and then process a command packet waiting in the Command Buffer.

4.4.1.2 Switches from Door & HV

The state of the Door may be determined by reading the MUX address. Some bits reflect whether HV and grid have been powered on.

4.4.1.3 Temps/Volts/Currents

An ADC converts sensor voltages to digital values to be stored in Housekeeping dataspace. The FSW must use the MUX to select the ADC input source, wait a short time, then read the ADC converted result and store it appropriately. Some tasks read certain inputs from the ADC at a high rate, while most inputs are read just before the FSW sends a Housekeeping packet.

The Current Limit Protection Task selects current sensors at the MUX and reads from the ADC to obtain a sample value used in the Protection algorithm. The Housekeeping Task reads all other selectable ADC sources and places them in the Housekeeping region.

4.4.1.4 Counters

Counters reflect the number of photon-event that passed through various stages of the digitizing process. The Fast Event counter is used by the Count Rate Protection Task to determine when a global count-rate violation has occurred.

4.4.1.5 PH-Histograms

Pulse Height (PH) Histogram data is 128 bins of consecutive 16-bit words in memory-mapped i/o space that represent the total number of photons detected, distributed into bins according to their pulse-height as determined by the detector digitizer. There are two PH-Histogram bins, one for MCP Segment A, the other for MCP Segment B.

4.4.1.6 Serial Port
The MON51 inspect-and-change task user interface requires the built-in 8051 serial port. The serial port driver is a not-for-flight part, so no traffic passes across this channel in a flight system. The inspect-and-change software component may remain part of the flight code in order to support laboratory testing and prototype compatability.

4.4.1.7 Timer (watchdog, other timers)
See 8051FC documentation for more information about the PCA timers internal to the 8051. The 8051 can be programmed to trigger an interrupt after a specified delay and can be used to generate time-delayed interrupt events.

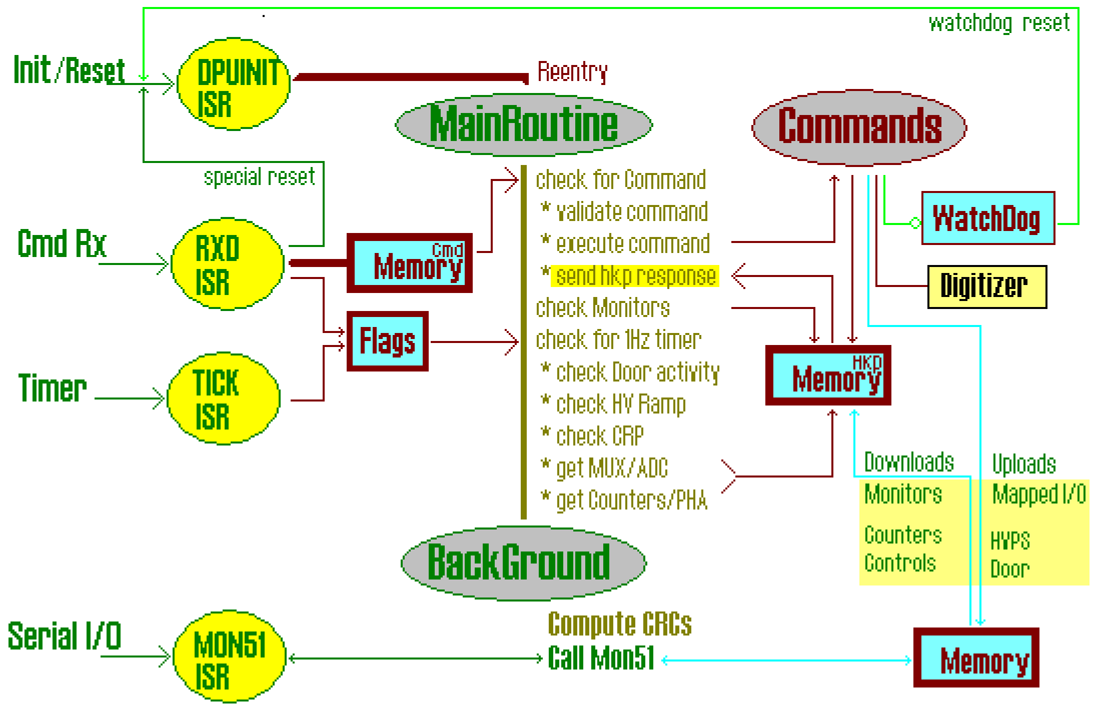

Figure 5: DCE Data Flow Diagram

4.4.2 Data Flow Events
The DCE Data Flow Diagram above depicts the data flow events that pass information among isrs, memory, and tasks. Events can originate externally, as with Commands, or internally, as with Timer-Ticks, or either, as with Resets.

4.4.2.1 Reset
A Reset event means that the DCE software finds itself executing code at address 0. The event is experienced by the 8051 very much like an interrupt generated in hardware.

4.4.2.2 Command Rx
When 32-bits of command arrives from the MEB, the DCE hardware (Actel, actually) generates an interrupt that triggers the Command Traffic ISR. The ISR stores the incoming 16-bit word to the specified 16-bit address. The address space available for writing command traffic is restricted to 0..0x1FFF. The upper three bits are reserved for special commands (TBD).

4.4.2.3 Timer Tick
The Timer Tick event originates with the internal 8051 timer function. The tick rate is programmed at any Reset or REENTRY to the mainroutine. The programmed rate is approx 49 ticks per second.

4.4.2.4 Door Timeout
The door Task performs a timed event using timer-tick events to force the door hardware to a known, protected state (Door Stop) in the event of a door command timeout, among others.

4.4.2.5 Door Latch
Another example of an event that forces Door Stop state. Whether the latch opens or closes, the FSW response is the same: Door Stop.

4.4.2.6 iLimit Shutdown,
An event that reports current overlimit events and may power-off hv and/or aux.

4.4.2.7 CRP Trigger
Excessive counts from the detector will force the HVLOW state. Default count rate and interval values may be changed on command. The CRP Task is a state machine that can force the HVState to HV_Low.

4.4.3 DATA FLOW SYSTEM STATES
In a ideal world, I'd have state diagrams – maybe even tables – for each of the system states below:

4.4.3.1 Boot/Operate Version States
(Boot version vs Operate versions in LCA & UCA)
The lines between state bubbles are events, such as JumpCode, Goto, WatchDog Reset, and POR Reset.

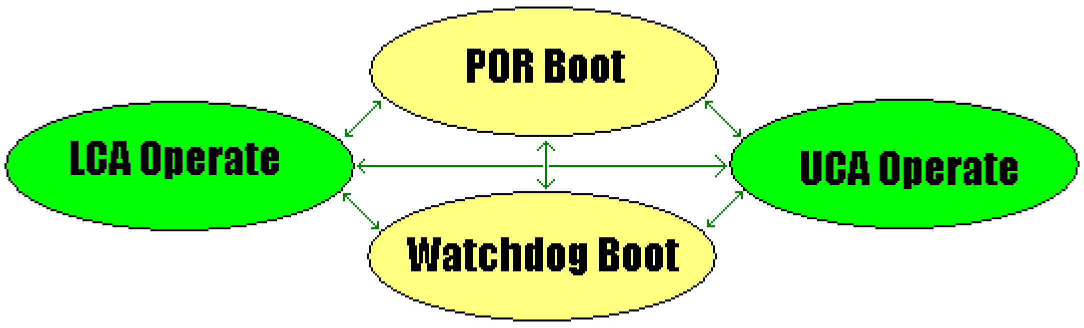

Figure 6: Boot/Operate State Diagram

4.4.3.2 HVState
(Off,On,Ramping,Low,Nominal,Unknown…)
The lines between state bubbles are events, such as HV Power on, HV Power off, HVState->Low, HVState->Nom, HV DAC Settings. The existence of the HV_MAX registers somewhat complicates the State Table. Omitted.

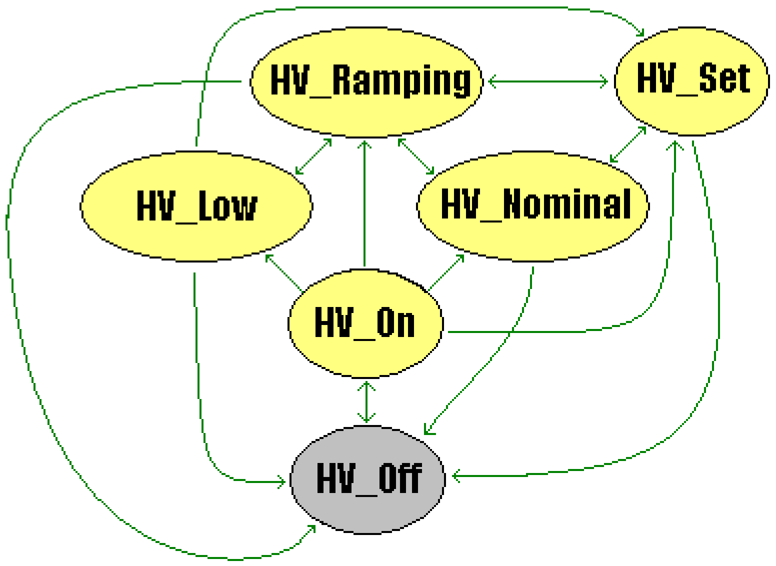

Figure 7: HV State Diagram

4.4.3.3 DoorState

(Open,Moving,Closed,Unlatched,Unknown)

4.4.3.4 Digitizer State

(Default POR, Programmed)

4.4.3.5 Diagnostic State

(Clear,POR,Watchdog reset,Error,Warning,Info)

4.4.3.6 Watchdog state

(Enabled/Disabled)

4.4.3.7 Count Rate Protection Task state

(Disabled, Sampling, Triggered, Waiting)
The state diagram below depicts the states the CRP Task may assume, as an example of the kind of state diagrams that show how a task functions. The drawing omits event names that should accompany each arrow indicating a state transition. See section 5.2.2.3 Count Rate Protection Task for a depiction of the task control flow that driven by the state of the Task.

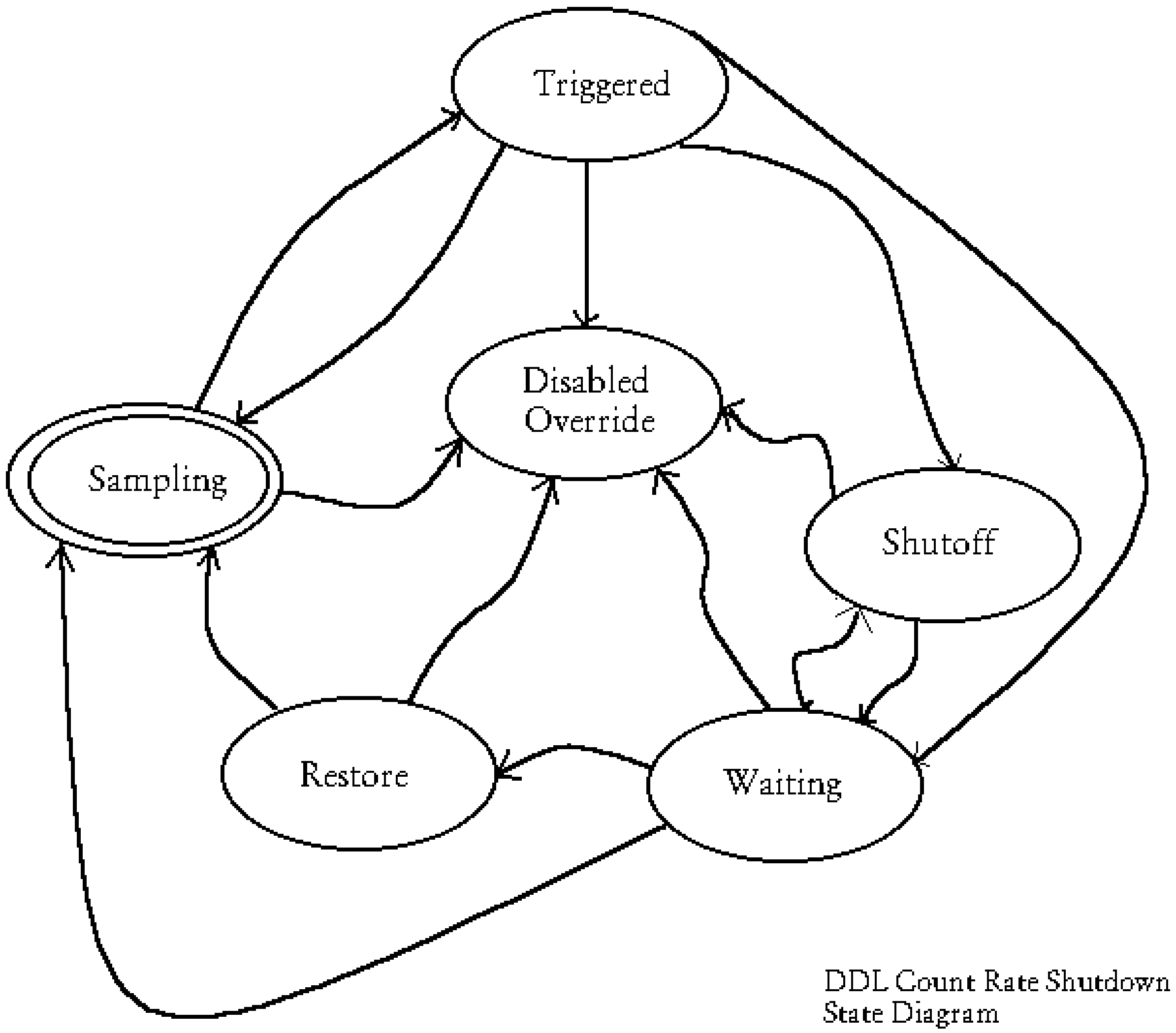

Figure 8: CRP State Diagram

4.4.3.8 Inspect-and-Change Task state

(Waiting-to-be-Initialized, Initialized, Handling SIO, Running a Command)
There is no data in housekeeping that indicates the state of this task. There is a variable that can be downloaded, but on flight hardware it will always indicate Initialized. A small number of variables are allocated to this task in the 32K-memory space.

4.4.4 DATA FLOW OUTPUTS:

Housekeeping, digitizer settings, door ctrl, HV ctrl, serial port, LEDs, reset pin

4.5 CONTROL FLOW DESIGN

The DCE FSW is a single threaded process and interrupt service routines that drive the states of several logical tasks. Always reset to Boot State, the software executes entirely from PROM data until commanded otherwise.

The basic design of software control flow is that of an initialization followed by an infinite processing loop. This control flow is originates with a reset event, such as power-on to the DCE. The main software processing loop, called the mainloop, can be interrupted by a hardware signal that directs the 8051 to execute a special sequence of code called an interrupt service routine (ISR). ISRs are designed to execute quickly and terminate, restoring control to the mainloop at the point it was interrupted.

4.6 DESIGN PHILOSOPHY

Data structures will be in fixed locations in memory and will generally not change from one version to the next. Command opcodes will be dedicated to the function and shall not be reused, conversely command function definitions for an opcode shall not change, although some commands may be retired and supporting code removed.

Commands for HV and Door are treated as special because they have the potential to “severely degrade” the performance of the detector. As special commands, the “energy” of the HV or Door will not be increased except when enabled by separate software interlock. This means that commands to turn off the door power, for example, do not require an “Enable” flag to execute the command; however, a command to turn on the door power, for example, will not turn on power unless a “Door Enable” command was previously issued. The general concept is that of sub-system “energy” level, where power applied to a subsystem has a higher energy level than the subsystem powered off; detector MCPs running at nominal HV has a higher energy state than the detector MCPs running at low HV.

5 DETAILED DESIGN

5.1 HARDWARE LAYOUT:

5.1.1 PORT I/O

The following functions have been assigned to pins at the 8051.

5.1.1.1 Self-Reset

The DCE 8051 is wired to wiggle it's own hardware reset line by triggering the power-on reset electronics. This capability is exploited during the initialization process to ensure that every hardware component that is connected to the hardware reset line gets properly reset on every reset event, regardless of source.

5.1.1.2 Serial Port

Used only to support inspect-and-change task communications with the host MON51 utility.

5.1.1.3 Digitizer Command Channel

For each digitizer, a 3-wire interface is implemented using 8051 port pins, bitbanging the data, gate, and clock bits to command the digitizer to accept a programmed value in a register internal to the detector digitizer.

5.1.1.4 PH Disable

The Counter hardware component writes to a memory that contains PH bin data. That memory is also accessible to the 8051. An attempt to access the same memory by both 8051 and Counter hardware, simultaneously, may result in corrupted data. Counter design requires that the 8051 assert a PH disable signal while accessing the Counter PH memory and de-assert the signal when done.

5.1.1.5 Control Lock

Some controls expose the detector to the risk of "severe degradation" if accidentally changed. To decrease the possibility of a spontaneous change to the control hardware, the hardware requires that the Lock signal be asserted when accessing those hardware controls, and de-assert the signal when done.

5.1.1.6 Special Bank Memory Switching

Spare 8051 pins have been routed to the Actel to allow for special functions to be associated with asserting the pin. It is envisioned that the Actel will allow mapping of some parts of the 8K RAM to code/data architecture, allowing the possibility that the Command Traffic ISR could upload and start execution of Operate code without the use of mainroutine functions.

5.1.1.7 8051 Control Registers

Memory Bits are internal 8051 memory that holds flags used to pass signals among tasks and to record state indicators. Some bits are reserved because they were used in FUSE for functions that do not apply to the DCE (i.e. mask select bits). Some bits are used to implement the digitizer command link that programs the digitizer with various settings.

Registers are in 8051 memory and control the behavior of such features as the serial port, timers, and watchdog.

5.1.2 MEMORY-MAPPED READ REGISTERS

The hardware component that is actually read when the 8051 tries to read from memory mapped i/o is determined by the Actel device, which generates chip-selects depending on a combination of bus address and asserted port pins.

5.1.2.1 Counters

```
Counter A                     6000-60FF
Counter B                     6100-61FF   Not in GALEX
```

There may be several counters in this region, at least a Digitized Event Counter (DEC) and Fast Event Counter (FEC) for each segment. Reading the counters will clear them.

5.1.2.2 PHD

```
PH-A (slice 0)              6200-623F
PH-A (slice 1)              6240-627F
PH-A (slice 2)              6280-62BF
PH-A (slice 3)              62C0-62FF
PH-B (slice 4)              6300-633F   Not in GALEX
PH-B (slice 5)              6340-637F
PH-B (slice 6)              6380-63BF
PH-B (slice 7)              63C0-63FF
```

The slice regions must be over-written with zeros to clear them, else the PH histograms will accumulate.

5.1.2.3 Switches from Door, HV

```
Bits'n'Switches (read)      4000-40FF   An 8-bit register
```

These bit read the door status (endswitches, and latch) motor power indicator and HV power indicator.

5.1.2.4 ADC Digitized temps/volts

```
ADC fetch (read)            4700-47FF
```

The conversion coefficients vary from one sensor to the next.

5.1.2.5 Rx Registers for Command traffic

```
3-wire Rx (read)            5000-5300   New to GALEX/COS
```

When reading the MSB (0x5000), the DCE hardware (Actel, actually) restores the Busy line that was forced active when the command data was sent.

5.1.3 MEMORY-MAPPED WRITE REGISTERS

The hardware component that is actually written-to when the 8051 tries to write to memory mapped i/o is determined by the Actel device, which generates chip-selects depending on a combination of bus address and asserted port pins.

5.1.3.1 HV Settings, Max

```
HV DAC A Setting            4101
HV DAC B Setting            4103      Not in GALEX
HV DAC A MAX                4100
HV DAC B MAX                4102      Not in GALEX
```

5.1.3.2 Control Bits for Door, HV

```
CONTROLS 1                  4200-42FF   An 8-bit register
CONTROLS 2                  4300-43FF   An 8-bit register
```

5.1.3.3 MUX Select input to ADC

```
MUX select (write)          4000-40FF
ADC start (write)           4700-47FF
```

5.1.3.4 Tx Registers for Housekeeping traffic

```
3-wire Tx (write)           5000-5300   New to GALEX/COS
```

Writing to the LSB starts the transmission of the four bytes stored in these registers.

5.1.3.5 LEDs

```
LEDS                        4E00-4FFF   Moved from FUSE
```

Not for flight, but a useful visual cue for aliveness in the lab.

5.1.4 RAM MEMORY

Memory -- nearly all memory in the DCE is statically allocated. The pigeonhole principle applies: every memory object has a dedicated location in DCE memory. Previously allocated memory locations are generally not used again. Violation of the pingeonhole principle has been shown to be the source of bugs and other adverse version-state dependent behavior.

5.1.4.1 Contents of Low 8K RAM

```
RAM data                      0-3FFF
RAM code, prom off            0-3FFF
```

Depicted in Figure 2, Memory Map.

5.1.4.1.1 Upload Buffer

Stores upload data passed as command traffic to this 1K buffer.

5.1.4.1.2 Command Buffer

Stores incoming command traffic in a small region of this 1K buffer.

5.1.4.1.3 Download Buffer

Data is copied to this buffer in the event of a download command, which then copies the entire download buffer out the housekeeping channel as a special download housekeeping packet.

5.1.4.1.4 Housekeeping Buffer

Housekeeping allocates a memory location for upto 1024 bytes of data. Housekeeping memory is the source of housekeeping traffic data and address tags. Many housekeeping memory locations will remain undefined in the FSW, but the Housekeeping task will always transmit the complete 512-word contents of memory. (In GALEX only, the housekeeping task will transmit only the housekeeping items that have changed since the last housekeeping packet.)

The Command Handler Task copied the small part of the command buffer that holds the command op-code and parameters to an image buffer in the Housekeeping region, and uses that image while executing the command.

5.1.4.1.5 Option to Remap Memory

If supported by the hardware…

5.1.4.2 Contents of High 32K RAM

5.1.4.2.1 Lower Code Area

LCA is defined as the address space from 0x8800 through 0xBFFF. Depicted in Figure 2, Memory Map.

5.1.4.2.2 Upper Code Area

UCA is defined as the address space from 0xC800 through 0xFFFF. Depicted in Figure 2, Memory Map.

5.1.4.2.3 Variables

Memory at address 0x8000 through 0x87FF are reserved for variables used by certain tasks. The variable space is used for data structures that are too big or not appropriate for storage in the Housekeeping data region. Depicted in Figure 2, Memory Map.

5.1.5 PROM MEMORY

ROM code, prom on 0-7FFF   0-3FFF
Depicted in Figure 2, Memory Map, in gray.

5.2 SOFTWARE LAYOUT

The software design has three major components: Reset initialization, Mainroutine, and Interrupt Service Routines.

5.2.1 RESET INITIALIZATION

Two kinds of resets are defined in the DCE FSW: Power-On Reset (POR) and Watchdog reset.

Reset Initialization distinguishes between POR and any other reset in order to determine how much internal memory to initialize. The watchdog reset should restore control to the boot-state mainroutine while preserving any evidence of the source of the reset event. Some initialization is required any reset to ensure control flow is fully deterministic -- i.e. we avoid the continuous-reset bug by ensuring that all variables we depend upon are properly initialized. To avoid the risk that a continuous-reset bug could hang the machine, there should be a watchdog reset counter that forces a POR after 255 reset events.

5.2.1.1 Power-On Reset (POR)

A power on reset is treated as if the hardware has just been powered on. While this kind of reset is intended to be used only during an actual power-on event, it is also a commandable event that forces the DCE into a default, known state. A watchdog reset assumes that power has been previously applied to the DCE, and that the reset event can have any source, including commanded reset, SEU, coding error, or other spurious event.

The POR initialization performs an "everything-but" initialization: stack filled with zeros, all dataspace below address 0x8000 set to zero, default digitizer settings sent to digitizer, door stopped, HV to HV_OFF state. Interrupts are enabled during POR, allowing the Command ISR to process incoming command traffic even before the Command Handler Task is ready to check the command buffer.

The POR purposefully does not write to memory addresses above 0x8800 in order to preserve the state of that memory. During a POR test, the watchdog reset and hardware resets are tested.

5.2.1.2 Watchdog Reset

This is a catchall-type reset that is named for its most likely cause: the 8051 built-in watchdog reset timer, which can cause the 8051 to jump to its reset vector. A reset can also originate as a coding error, command, or SEU in code.

The initialization performed after a watchdog reset does not include activities performed during POR, but does include reinitializing variables used in system tasks, and always forces the door to the stopped state.

The diagram below depicts the FUSE/DPU reset routine, shown here because the COS/DCE reset routine will be of similar design. The parenthesized comments in the diagram indicate messages that are passed out the SIO channel.

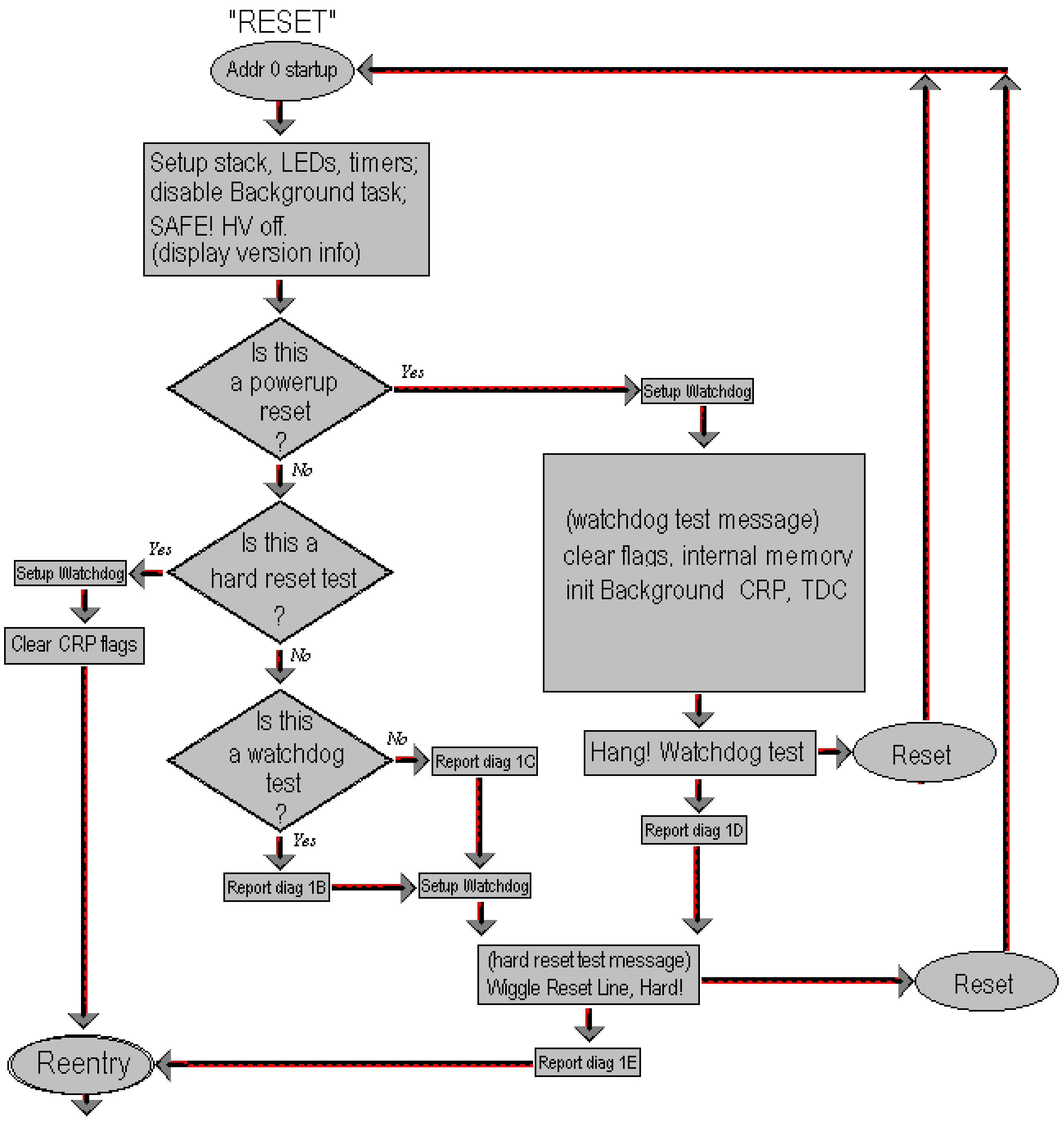
"RESET"
Addr 0 startup
Setup stack, LEDs, timers; disable Background task; SAFE! HV off. (display version info)
Is this a powerup reset ?
Yes
No
Setup Watchdog
(watchdog test message) clear flags, internal memory init Background CRP, TDC
Is this a hard reset test ?
Yes
No
Setup Watchdog
Clear CRP flags
Is this a watchdog test ?
No
Yes
Report diag 1C
Hang! Watchdog test
Reset
Report diag 1D
Report diag 1B
Setup Watchdog
(hard reset test message) Wiggle Reset Line, Hard!
Reset
Report diag 1E
Reentry

Figure 9: Reset Initialization Algorithm

5.2.2 MAINROUTINE

Following reset initialization, control passes to the system mainroutine. The mainroutine consists of an entry point (named REENTRY) initialization followed by a mainloop that calls each of the system tasks in round robin fashion. There are no timing delays built into the mainloop.

The flow diagram below depicts the mainloop and its tasks.

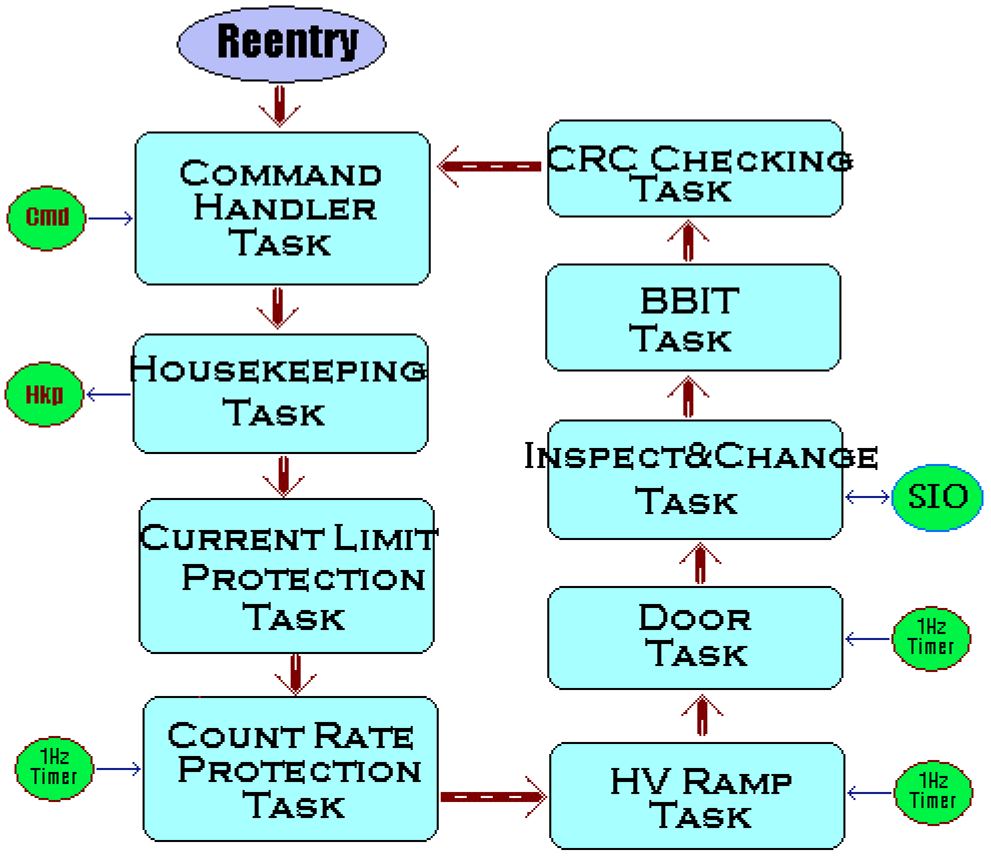

Figure 10: Mainroutine Tasks

The DCE mainloop can be considered a single-threaded process that contains several tasks. Tasks are designed as routines that a called repeatedly, executing a segment of code that may change the tasks state and stores all its variables for use next time the routine is called. Tasks should execute their routines quickly in order to return control to the mainloop. There is no RTOS in the DCE. The passing of data and control among the tasks is controlled by the mainloop.

System Tasks include the Command Handler Task, an inspect-and-change monitor (not for flight use), a Count-Rate Protection Task, Current-Limit Protection Task, Background CRC checking task, Background Built-in Test (i.e. stack-check), the Housekeeping Task, HV Ramp Task, and Door Task. Unlike tasks used by a RTOS, the tasks in this system are interleaved in the mainloop, which explicitly calls the code for each task in turn. Similarly, tasks may not be created or destroyed in the DCE FSW system.

Their well-defined states, functions, subroutines, and associated data structures characterize tasks. Tasks may share subroutines and pass signals through shared data structures. Task code is constructed so that it's state changes over the course of repeated calls to the task. The task code should perform a quick code execution and return control to the mainloop. Tasks do not hog CPU cycles, but rather they perform some portion of an algorithm that depends on repeated calls to the task. Tasks store their state variables at a dedicated location in housekeeping memory. The mainloop calls each task in turn as quickly as possible, interrupted only by a timer tick and incoming command traffic.

5.2.2.1 Command Handler Task

The Command Handler Task detects an op-code in memory (placed there by the Command Traffic ISR); checks for a valid format and then executes the command. The command itself will write results to housekeeping as appropriate. The Command Handler Task also sets a flag used by housekeeping task to indicate that a housekeeping response is required. When the DCE is in Boot-State, some commands will not be available, namely the commands that enable hardware functions to the HV and Door.

See the ICD for a complete list of commands and parameters.

5.2.2.1.1 Inputs
Command traffic is written to the Command Buffer as is arrives.

5.2.2.1.2 Events
While polling the memory location of the Command packet op-code, the Task recognizes the arrival of a new op-code.

5.2.2.1.3 States
The command handler copies the contents of the command buffer to a command image in the Housekeeping region, thus preserving the parameters for use by the command. When the task returns control to the mainroutine, it is always in the same state: searching for a command opcode.

5.2.2.1.4 Functions
Increments the command packet counter
Checks for Valid command format
Increments the command received counter
Performs the command using specified parameters.
Increments the command executed counter

5.2.2.1.5 Outputs
The command handler may produce a housekeeping packet that contains download data in response to a download command. Always sets the signal to the Housekeeping task to transmit a Housekeeping packet in response to the command, valid or not. Produces the action specified by the command.

5.2.2.2 Inspect and Change Task

Inspect and Change Task is a modified Archimedes MON51 utility. The program uses the 8051 serial port to communicate with a PC running the MON51 host software. The serial port is not for flight, so this task does nothing on flight hardware, except consume processor cycles. However, this task is useful when run on breadboard or prototype hardware and the programmer wishes to view the contents of DCE memory. Some MON51 functions do not work -- only inspect & change of Data space is supported by this task.

5.2.2.2.1 Inputs
At initialization, a state variable is set to force the inspect-and-change code to initialize. A PC-based utility, MON51, sends queries over the serial port.

5.2.2.2.2 Events
Incoming serial port activity wake up the task to process the query

5.2.2.2.3 States
Unknown. May require several calls

5.2.2.2.4 Functions
See the MON51 manual for details. Some commands don't work.

5.2.2.2.5 Outputs
The command handler may produce a housekeeping packet that contains download data in response to a download command. Always sets the signal to the Housekeeping task to transmit a Housekeeping packet in response to the command, valid or not. Produces the action specified by the command.

5.2.2.3 Count Rate Protection Task

Count Rate Protection Task checks Digitizer counters on a regular basis and forces HV to the HV_LOW state when excess counts over some interval are detected. This task asserts direct control over the HV State when a Count Rate limit is exceeded, and may change the state of other tasks.

5.2.2.3.1 Inputs
Default parameters are used at POR, and may be modified on command. Every second, this task reads the counters and compares accumulated counts against in table of counts in the variable memory region.
Commands that change the HV State will also reset the state of this task to Sampling.

5.2.2.3.2 Events
Every one-second-timer tick is the signal to check the counters against the limit table. If the accumulated counter exceeds any member of the table, then the protection function is triggered. When triggered, this task may optionally resume from the protected state after a specified delay.

5.2.2.3.3 States
Disabled, Sampling, Triggered, Waiting are the states applicable to COS/DCE, which by default will not autonomously resume from the triggered state. See the diagram below, which was made for FUSE, for an example of a state diagram.

5.2.2.3.4 Functions
See the diagram below, which was made for FUSE, for an example of a flow control diagram for a task.
Checks count rate activity and when triggered, forces the HV state to HV_low.

5.2.2.3.5 Outputs
Updates a table in variable memory. May command the HV state machine.

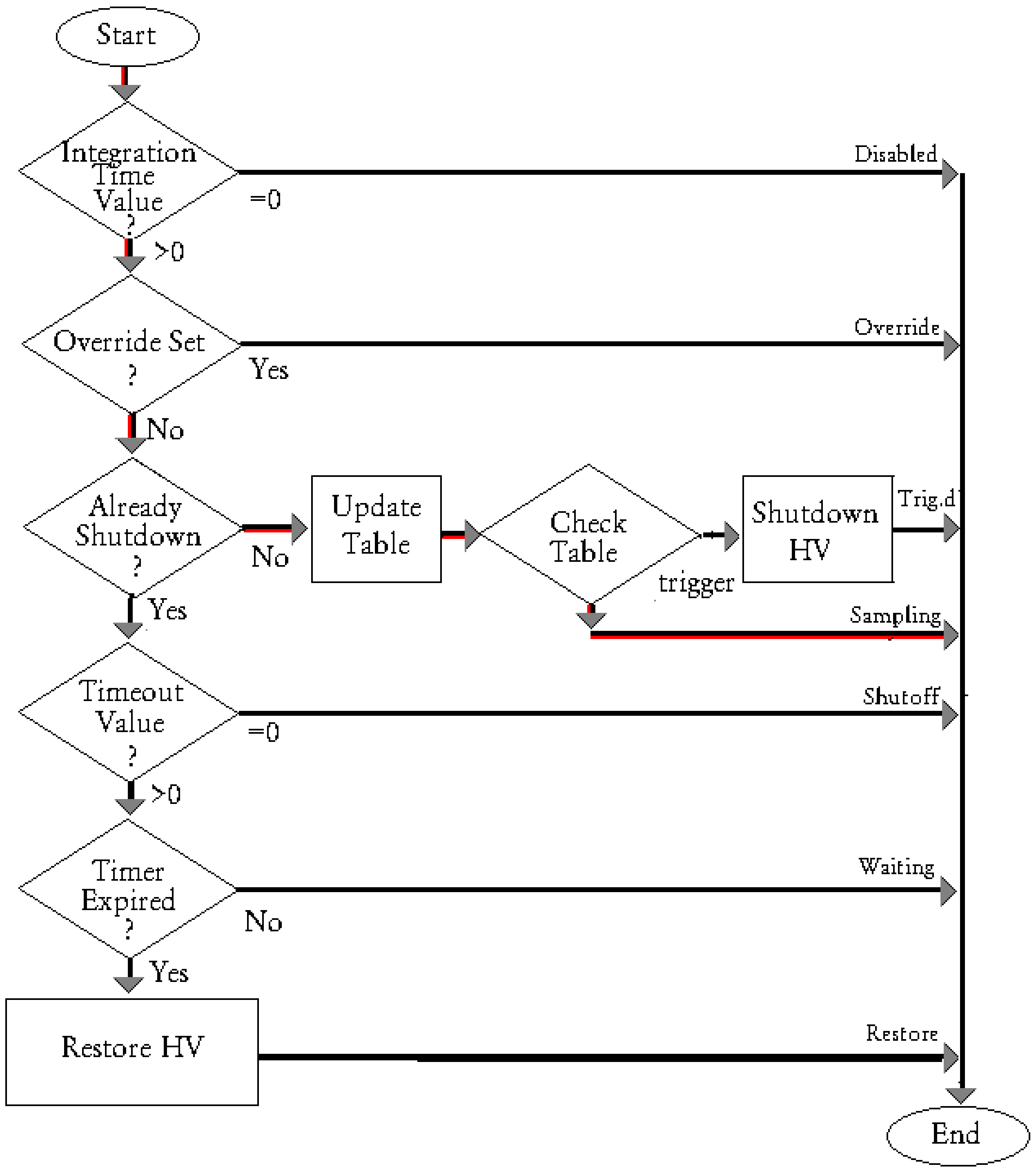

Figure 11: CRP Task Algorithm

5.2.2.4 Current Limit Protection Task

Current Limit Protection Task checks HV and Aux current readings, records samples, and in the event of an overlimit event will report a diagnostic and possibly turn off HV.

5.2.2.4.1 Inputs
Default parameters are used at POR, and may be modified on command. As fast as possible, this task reads the ADC readings for HV current and Aux current. Commands that change the HV State will also reset the state of this task to Sampling.
5.2.2.4.2 Events
If a single overlimit event is detected, then this task reports a diagnostic; the next several overlimit samples are ignored, but the state machine is reset if a sample is not overlimit; after some number of sequential overlimit samples, then the protection function is triggered. When triggered, this task will turn off power to the HV supply. There is no autonomous resume from the protected state.
5.2.2.4.3 States
Disabled, Sampling, Triggered, Waiting are the states applicable to COS/DCE.
5.2.2.4.4 Functions
Reports HV glitches without power-off of HV
Recognizes a sustained overlimit condition, and in the event will power-off HV and report the overlimit event.
5.2.2.4.5 Outputs
Saves samples in RAM, available by download command.

5.2.2.5 CRC Checking Task

Background CRC checking computes a 16-bit value that characterizes a defined region of DCE memory. The task reports changes to a memory region when the CRC computed on that region differ from a previously computed value. This task reports changes as diagnostic reports and does not attempt to correct any memory errors.

5.2.2.5.1 Inputs
Initialized to point to the first in a link-list of data structures specifying memory regions to be checked. Some commands may also reinitialize the CRC-checking machine.
5.2.2.5.2 Events
Reset and some commands will force the CRC-checking machine into a specific state. In general, the task completes its check of some region and automatically starts computation of the next region in the linked list.
5.2.2.5.3 States
Task is continuously computing the value of one of a set of memory regions included in the linked list. If the region being scanned represents the state of the task, then this task has as many states as there are regions in the linked list.
5.2.2.5.4 Functions
Processes one byte of data through the CRC algorithm every time the task is called.
When the region has been computed, the task checks to see that the value matches an image in memory and reports a diagnostic when a new value is stored in the housekeeping record.
5.2.2.5.5 Outputs
Computed CRC values have designated locations in housekeeping memory.

5.2.2.6 Background Built-in Test

Background Built-in Test performs a number of sanity checks on the DCE and reports anomalies as diagnostics. One identified BBIT is a check on the 8051 stack that reports when an unexpected amount of stack space is used. This is necessary because the stack is the only dynamically allocated memory in the DCE system. Another BBIT is the watchdog timer, which is serviced in the Command Handler Task.

5.2.2.6.1 Inputs
Initialized high-water mark to some value that doesn't trigger an event in normal operations. Reads internal 8051 registers, possibly other housekeeping items...
5.2.2.6.2 Events
Stack incursion detected.
5.2.2.6.3 States
High-water mark recorded in memory.

5.2.2.6.4 Functions
Reports stack incursion as diagnostic.
5.2.2.6.5 Outputs
Generates diagnostic and records new mark in the event of a stack incursion event detected.

5.2.2.7 HV Ramp Task

HV Ramp Task provides control of the HV ramping process and possibly checks for expected HV responses. This task interacts with CRP, CLP, and the Command Handler Task.

5.2.2.7.1 Inputs
HV Ramping is initiated on command.
5.2.2.7.2 Events
Command turns on ramp flags.
CRP and CLP can stop ramping or even force HV to the off state.
5.2.2.7.3 States
Ramping or not.
5.2.2.7.4 Functions
Increments the HV setting to toward a command value.
5.2.2.7.5 Outputs
Task writes to HV DAC registers. Does not touch HV MAX registers.
Turns off ramp bits when HV reaches commanded value.

5.2.2.8 Door Task

Door Task provides checks on the door timer and latch events, and, in the event, executes a DoorStop function.

5.2.2.8.1 Inputs
Commands to control the door.
5.2.2.8.2 Events
Latch moved, detected at MUX 8-bit register, indicates we should stop the door.
Any reset will stop the door.
Three minutes after the last door command, the door will be autonomously commanded to stop.
Aux current overlimit will stop the door.
5.2.2.8.3 States
Aux power off, DoorStoppedSomewhere DoorStoppedOpen, DoorStoppedClosed, DoorEnabledSomewhere, DoorEnabledOpen, DoorEnabledClosed, DoorOpening, DoorOpen, DoorClosing, DoorClosed. All these states are detected and reported in housekeeping. After approx. 3-minutes, this task forces its state to one of the Door Stopped states.
5.2.2.8.4 Functions
Primarily used to stop the door in any of several ways.
5.2.2.8.5 Outputs
Door stop. The reason why is reported as a diagnostic.

5.2.2.9 Housekeeping Task

Updates housekeeping data space and sends a housekeeping packet on signal.

5.2.2.9.1 Inputs
A bit signal to collect and to start transmission of housekeeping data.
5.2.2.9.2 Events
Command Task sends the bit signal
5.2.2.9.3 States
Waiting to Send/Sending

5.2.2.9.4 Functions
Collects housekeeping data and transmits a housekeeping packet to the CS.
5.2.2.9.5 Outputs
Housekeeping data space updated, housekeeping channel passes a packet of 512 32-bit housekeeping words.

5.2.3 INTERRUPT TASKS

5.2.3.1 Timer Interrupt Service Routine

The timer interrupt starts an ISR that sets flags used by some tasks to control timing behavior. For example, the CRP algorithm checks for a limit violation just once per second. The CRP Task may get called several times per second, but performs its timed functions only when signaled by a flag set in the Timer ISR.

5.2.3.1.1 Inputs
Programmed at initialization to enable the interrupt and trigger an interrupt on a regular basis, say 49 times per second.
5.2.3.1.2 Events
There is a regular 8051 timer interrupt, and also a set of internal 8051 timer registers, which when matching the PCA timer, will generate programmable interrupts. At least one is used by the watchdog.
5.2.3.1.3 States
Running/Not Running: the DCE could perform without this ISR, but some timed activities wouldn't function properly. It is also possible to program the Timer for a variety of intervals.
5.2.3.1.4 Functions
Provides timer-tick signal to tasks that perform timed activities.
5.2.3.1.5 Outputs
Sets 8051 register bits to indicate timing events, such as one second.

5.2.3.2 Command Traffic Interrupt Service Routine

The special features of the Command Traffic ISR are intended to provide emergency access to the DCE and will not be used as the primary method of uploading executable code. The command traffic ISR drives a state machine much like a task, but is not called by the mainloop. The ISR's main function is to move data into memory for analysis and processing by the command handler task. The ISR also provides a means to reset the software if the mainloop should hang for any reason. If supported in hardware, the ISR could be used to transfer code image and start it running.

The command traffic interrupt starts and ISR that collects the 32-bit word and processes it as follows:

```
if the MSB is less than 0x0800, then the 32-bit word is probably a memory
  transfer, so the lower 16-bits are stored at the address specified by
  the upper 16-bits.
else if the MSB equals 0x8000 and all the other bytes are 0, then we
  recognize this as a special reset command that is executed immediately
  from within the ISR.
else if the MSB equals 0x8001 and all the other bytes are 0, then we
  treat this as a special interlock that enables special control commands.
  The interlock is reset when a housekeeping packet is sent.
else if the MSB equals 0x8002 and the interlock is set, then the ISR
  immediately passes control to the address specified in the lower 16-
  bits.
else if the MSB equals 0x8003 and the interlock is set, then the ISR
  toggles a banked-memory switch that maps a portion of the RAM to code
  space.   This function provides the hook required to allow code to be
  uploaded and executed using the Command Traffic ISR alone.
```

5.2.3.2.1 Inputs
At initialization the interrupt is enabled. Anytime DCE hardware receives a 32-bit command word, the hardware (Actel, actually) generates an interrupt the calls upon the ISR to perform some sequence like the algorithm above.

5.2.3.2.2 Events
Incoming 32-bit command words are handled completely in the ISR

5.2.3.2.3 States
Data transfer is pretty much a single-state machine, except this ISR provides for a backdoor means of forcing a reset, and may also provide for uploading code and starting its execution. Command packets are not interpreted by this ISR.

5.2.3.2.4 Functions
Supports flow of command traffic into Command Buffer memory. Provides a backdoor upload & operate scheme. Provides a POR command.

5.2.3.2.5 Outputs
When in the Data Transfer State, this ISR writes to RAM in the Command Buffer region.
May transfer control on command. May trigger a POR.

6 HARDWARE AND SOFTWARE ENVIRONMENTS

The following sections summarize the hardware and software environments in which the DCE FSW will reside and operate. Also listed are software tools that are part of the development environment.

6.1 HARDWARE ENVIRONMENT

The following hardware items are the major hardware components that together form the hardware environment in which the DCE FSW will reside:

ITEM	DESCRIPTION	COMMENTS
Microprocessor	UT69RH051 (rad-hard 8051FX)	16 MHz
PROM	16 Kbytes	Boot code and operational code image
RAM	32 Kbytes	Operational code image and data
Low RAM	16 Kbytes	Command and Housekeeping data, including data transfer areas; may contain interrupt service code

6.2 SOFTWARE ENVIRONMENT

The following software items are the software packages that together form the software environment in which the DCE FSW will be developed and reside:

ITEM	DESCRIPTION	COMMENTS
Configuration Manager	Pkzip	Every tested version saved
8051 Assembler	Archimedes A51	Archimedes
Linker	Archimedes L51	Archimedes
Object-to-Hex	Archimedes OH51	Archimedes
In-Circuit Emulator	High-Tex ICE	High-Tex
Hex2cmd	Creates upload script	EAG
Hexmv	Moves load addresses to PROM space	Required for creating PROM burnable code images.

7 ASSORTED NOTES

7.1 DCE FSW REQUIREMENTS MATRIX

5.1.1.1	"Initialize to Boot State After Reset"
5.1.1.2	"Indicate The Cause Of The Reset"
5.1.1.3	"Commands To Reboot DCE FSW"
5.1.1.4	"Memory Initialization On Power-Up Only"
5.1.1.5	"Code In PROM (Non-Volatile Memory)"
5.1.2.1	"Watchdog"
5.1.2.2	"Command to Disable/Enable Watchdog"
5.1.2.3	"Capability To Log Diagnostics"
5.1.2.4	"HST Error Logging"
5.1.2.5	"HST Error Format"
5.2.1.1	"Error-Detecting Command Format"
5.2.2.1	"Command Rate: One Per Second"
5.2.3.1	"Housekeeping Response Within One Second"
5.2.3.2	"Echoes For Command Opcode And Parameters"
5.2.3.3	"Counters For All Commands"
5.2.3.4	"PHD Handling"
5.2.4.1	"Command to Jump Anywhere in Code"
5.3.1.1	"HV Ramping Function"
5.3.1.2	"Commands To Set Detector Digitizer Gains And Offsets"
5.3.1.3	"Commands To Control High Voltage"
5.3.2.1	"Commands To Enable/Disable Global Rate Monitoring"
5.3.2.2	"Commands To Set Global Rate Monitoring Parameters"
5.3.2.3	"HV Current Over-Limit Checking"
5.3.2.4	"HV Interlocks"
5.3.2.5	"Take Appropriate Protection Measures If Limit Exceeded"
5.3.2.6	"Command Enable Or Disable Limit Checking"
5.4.1.1	"Door Direction"
5.4.1.2	"Door Power"
5.4.1.3	"Actuator Power"
5.4.1.4	"Door Override"
5.4.1.5	"Settable Door Current Limit"
5.4.2.1	"Door Timeout"
5.4.2.2	"Door Enable"
5.4.2.3	"Aux Current Checking During Door Motion"
5.4.2.4	"Latch Checking"
5.5.1.1	"Process Memory Uploading Commands"
5.5.1.2	"Capability To Upload Data"
5.5.1.3	"Check Upload Integrity"
5.5.1.4	"Command To Disable/Enable CRC Checking On Upload"
5.5.2.1	"Process Memory Download Commands"
5.5.2.2	"Download Data Timing"
5.5.3.1	"CRC Background Checking On Memory Regions"
5.5.4.1	"HST Memory Monitors in Housekeeping"
5.5.4.2	"HST Memory Monitor Commanding"

7.2 MISC

For each requirement above, the matrix shows which ISR or Task is allocated responsibility.
Responsible modules are named as follows:
- Initialz
- Command ISR
- Timer ISR
- Mainloop
- CRP Task
- CLP Task
- CRC checking Task
- BBIT (latch check) Task
- HV Ramp Task
- Door Task (includes latch check and timeout)
- Housekeeping Task
- and Common Subroutines