Revision

A

SCIMEASURE ANALYTICAL SYSTEMS

The "Little Joe" Digital Camera

# Technical Manual and Users' Guide

# Table of Contents

THE "LITTLE JOE" DIGITAL CAMERA

# Technical Manual


1123 Zonolite Road • Suite 28
Phone 404-876-6558 • Fax 404-876-6004
Web www.SciMeasure.com

Chapter 1

# Overview and Features

*The purpose of this manual is to acquaint the user with the functionality and overall design concept of the Little Joe camera.*

Little Joe is a CCD based scientific camera. The primary mission of Little Joe is to provide digital video at high frame rates and low light levels. Its intended user is a scientist, such as an astronomer or biological researcher. Although significant effort has been made to make the camera easy to use, the main design emphases are on performance and versatility. There are, of course, many CCD camera designs extant in the consumer and scientific domains. Consumer CCD camera designs are generally driven by ideals of attractive appearance and acceptable performance, whereas scientific CCD cameras are generally designed with a specific application in mind and support a limited number of CCDs in a single form factor. Even many application-specific scientific CCD camera designs make design compromises for the sake of standardization.

Consumer grade CCDs and CCD cameras typically strive to deliver the highest resolution image at an acceptable visual quality. Scientific CCD cameras are typically designed to minimize read noise at a desired readout rate while maximizing dynamic range.

SciMeasure's Little Joe is extensible to all known CCDs by design, while allowing the highest possible performance of a CCD to be realized. Although the camera can stand alone, and can support a variety of output formats, it is also important that it can be computer-controlled and can provide calibrated digital data for analysis. Little Joe's performance specifications are due in large part to its versatility, which in turn is due to the camera being modular, configurable, and programmable.

*Modular*

The fundamental property of the design is the CCD Controller Bus. The bus incorporates all the features required to control all known CCDs. These include a Digital Power Bus, an Analog Power Bus, an $I^2C$ Serial Bus, an RS-232 Serial Bus, a Digital Sequence Bus and an Image Data Bus. These features are described in more detail in Chapter 2 (Theory of Operation).

Although the Controller Bus architecture enables an arbitrary number of camera designs for a variety of CCDs, certain functions must always be performed. In order to achieve the highest possible performance from CCDs, a suite of modules exploits their separation of functionality while optimizing the performance of the camera as a whole. The individual modules are discussed in depth in Chapter 3 (Module Descriptions).

## *Configurable*

While modularity is key to optimization and versatility, it also provides the useful by-product of making the camera configurable to customer-specific applications. To help understand what we mean by this, here are some examples from actual customer case histories:

- A standard Little Joe configuration provides the image data in a continuous "rasterized" stream to ease image acquisition. Customer A wanted, in addition to that, the raw unmultiplexed data from a multiple output CCD in order to perform real-time analysis. The data had to be presented in both formats simultaneously. Configuring Little Joe with two Output Modules, each providing one of the formats, fulfilled Customer A's requirements.

- A normal tradeoff in analog digitizers is dynamic range *versus* speed. The more range, the less speed, and *vice versa*. However, Customer B required both high speed and a large dynamic range — not necessarily at the same time, but in the same camera, so that the subject could be observed both ways without changing the setup. His Little Joe was provided with two sets of Input Modules: one providing the range, the other providing the speed. The observational mode can be selected by a mouse click — no unplugging, re-cabling, shutdown or stoppage is required.

- Customer C has several Little Joes that must operate at different frame rates, but nevertheless in synchrony. A new, custom synchronizing module was developed to attach to the bus and provide this functionality with no other rework of the system.

## *Programmable*

Little Joe's versatility is further enhanced by its programmability. Programming will fall into one of three general categories: image acquisition, camera functionality, or camera operation.

- **Image acquisition**. Fundamental to any CCD based camera is the set of signals (waveforms) that control the CCD in its generation or acquisition of the image. These waveforms not only determine such things pixel rate, frame rate and frame size, but also can have enormous effects on image quality in terms of noise and responsivity. In Little Joe, the waveforms can be programmed either at the factory or by advanced users.

- **Functionality**. Many of Little Joe's modules contain programmable logic devices or microcontrollers that determine how the camera operates apart from image acquisition. New features can be added, or errors corrected, by simple reprogramming (at the factory, not by the user).

- **Operation**. The Users' Guide describes a set of commands for controlling or programming the operation of the camera. Most operating parameters or modes can be programmed to persist through a power cycle, so that setup work need not be lost or repeated unnecessarily.

Chapter 2

# Theory of Operation

*A high level description of the Little Joe camera is given, to aid in developing applications and using the camera to its potential.*

A Charge Coupled Device (CCD) is a semiconductor imaging device that is essentially an array of photo-sensitive capacitors controlled by a grid of wires. Bias voltages are used to power the device and clock voltages are used to move the charge around on the device. Frame transfer CCDs simply have an image array and one or more serial registers. The image accumulates in the image array, as photo-electrons are generated. A shutter is generally employed to prevent streaking while the image is transferred to the serial registers. From these registers, pixels are shifted to an output driver, one pixel at a time. Frame storage CCDs are similar, with the addition of an (image) frame store, which lessens the requirement for a shutter.

A CCD camera is generally comprised of a CCD and its controller. They are frequently housed in the same enclosure, especially in consumer applications, but are frequently housed separately in high performance and specialty applications. The CCD controller provides the bias voltages, clock voltages and output drivers, and must clock the CCD in a manner that achieves image integration and readout. In the case of digital cameras, the analog voltages from the outputs must also be digitized.

The block diagram below gives an idea of how an image is generated in the Little Joe. A typical 4-port system is shown. Photons hitting the CCD are converted to an electrical signal by that device. The signal is amplified locally, then transmitted to the controller for further processing. As shown, the gain, frequency response and black level offset are manipulated before the signal is digitized by an Analog to Digital Converter (ADC) circuit. The digital information from multiple channels is then multiplexed onto the image bus. At this point, the image bus holds pixel data in a sequential order that is dependent upon the geometry of the CCD. From here the pixels are framed into an image (e.g. rasterized) and otherwise made suitable for digital transmission to the user's own image processing equipment.

The bottom portion of the diagram depicts how the signals that drive the CCD are generated. Under direction of the microcontroller (which is itself under user control via the serial port), the controlling sequence is fed into the RAM. It is translated to the necessary voltages by the clock drivers, and along with the bias voltages, sent to the CCD.

The 6 bit control bus shown leaving the RAM is synchronous with the clock sequence and therefore operates at high speed; but it has certain limitations. The serial control bus is low speed and asynchronous, but much more versatile in its capabilities.

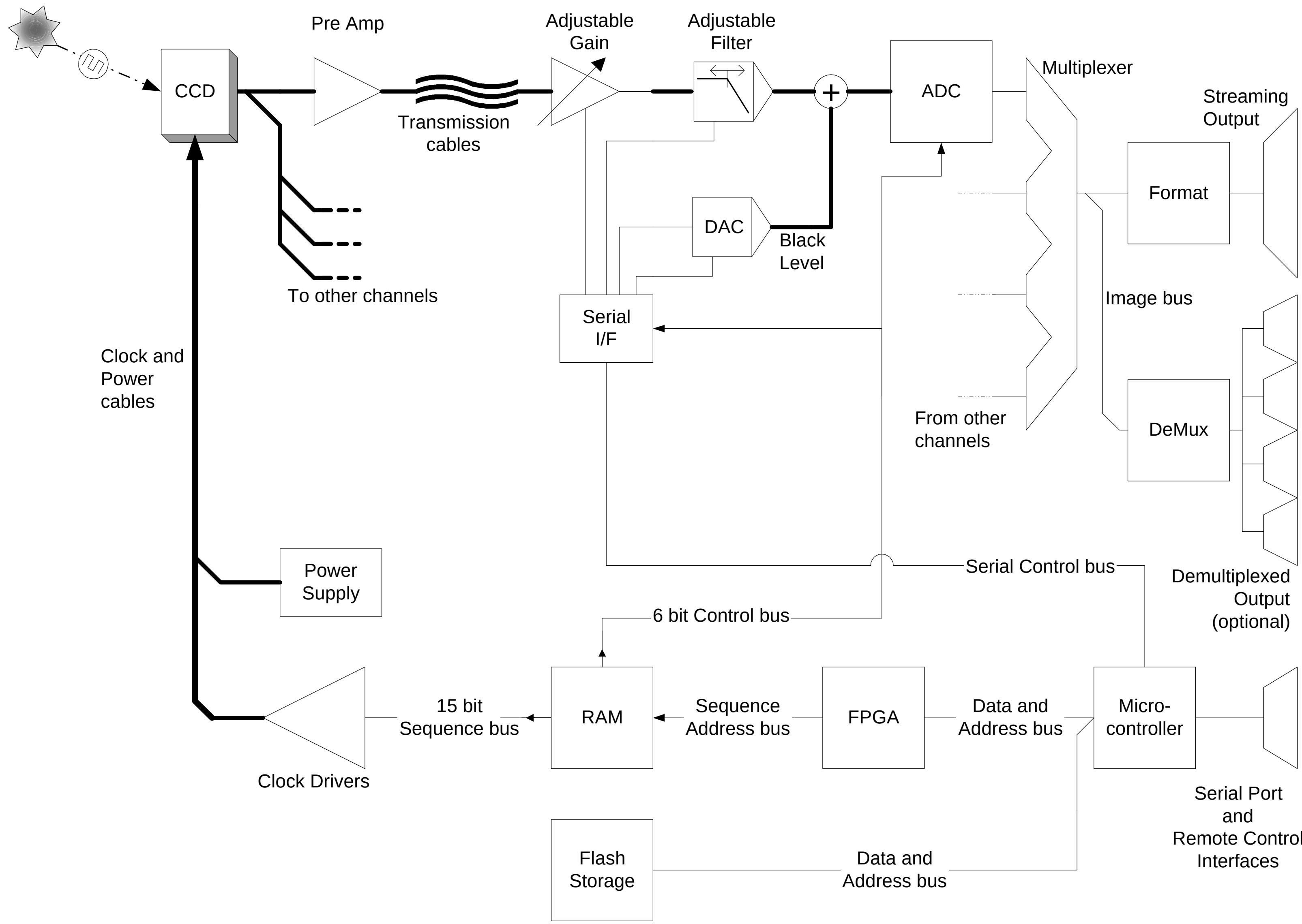



**FIGURE 1.** Block diagram of the Little Joe . *The darker lines depict analog voltages and signals, while the rest are digital. No attempt is made to depict modularity. See text above for a more detailed description*

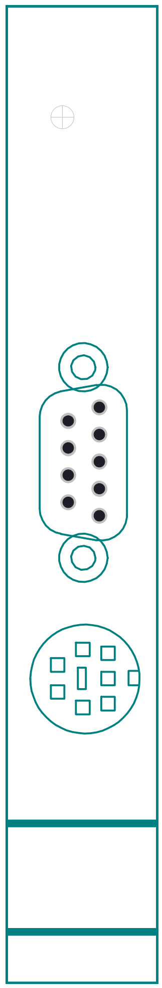
Chapter 3

# Module Descriptions

*Here the modules currently available are discussed in a greater level of detail.*

- Command Module
- Service Module
- Clock Driver Module
- Camera Head
- Input Module
- Output Module
- Timing Module
- Backplane

# Command Module



The Command Module is the heart of the instrument and its operation is integral to the philosophy of the Controller Bus. By means of a microcontroller and an FPGA, the Command Module implements RS-232 and $I^2C$ serial interfaces, interfaces with fast static RAM and Flash RAM, and operates a high speed Digital Sequencer.

The RS-232 serial interface allows communication with the outside world and external control by means of the AIA Standard Protocol for digital cameras. The physical connection may be made through any module that supports it.

The $I^2C$ serial interface is used by the Command Module to control and query other modules on the Controller Bus. This is an industry standard simple two-wire bus that allows control of settings such as gain or offset, and reading the temperature or other status of any module in the system.

The Digital Sequencer resides in an FPGA and uses data in banks of fast static RAM to generate a sequence of digital patterns known as a Digital Sequence. These patterns are mapped to the CCD clock inputs, to dedicated signals such as Clamp and Sample, and to special control bits.

CCD cameras generally use state machine based sequencers that rely on sequential logic to generate the clock patterns, or DSP based sequencers that use software to generate the clock patterns on the fly. The former has the disadvantage of being impossible to program arbitrarily, and the latter has difficulties with speed and complexity. The RAM-based Digital Sequencer has the advantages that: 1) the contents of the RAM are completely arbitrary, and 2) they can be changed with ease. There is a limitation in terms of overall sequence length, but this is true of all sequencers. With over 1 Megabyte of storage available for bit patterns in the Little Joe, this limitation has yet to prove to be restrictive.

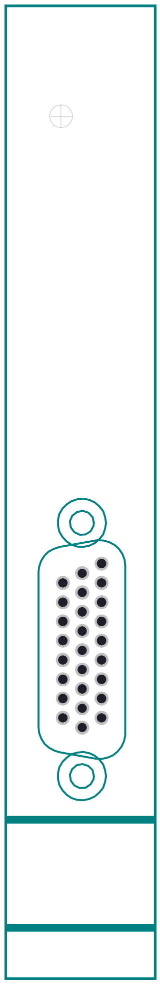
In our design, the bit patterns are clocked at 50 MHz resulting in a resolution of only 20 nSec for each signal.

There are two basic phases to reading an image out of a CCD. The first is image integration while the CCD is exposed to the image source, and the second is the readout. In frame storage CCDs, these phases can overlap substantially and there may not even be a separate integration phase. However, during a separate integration phase, the Sequencer can either stop clocking the CCD, or it can execute a special integration pattern. This is especially useful when a dithering technique can be used to minimize dark current within the CCD during long integration periods. The last pattern executed in each program is defined to be the integration pattern. A special accumulation register can be set to repeat the integration pattern up to 65536 times, giving extremely fine control over exposure times.

Program selection and synchronization of the controller to an external device, or *vice versa*, is often desirable in science. For this purpose, the Command Module provides this utility through a Remote Synchronizing Interface (RSI) connector. Signals indicating Start of Sequence and Start of Integration are provided as TTL outputs and a RUN signal is input as a TTL signal. In addition, three program selector bits are provided as TTL inputs to allow program selection by an external device "on-the-fly". More detail on these features is contained in the Users' Guide "Hardware Interface" chapter.

Sequences can be uploaded to the controller arbitrarily through the RS-232 Serial Port, but they can also be stored in Flash RAM. This is also true of the various settings within the controller, such as gain or offset.

## Service Module

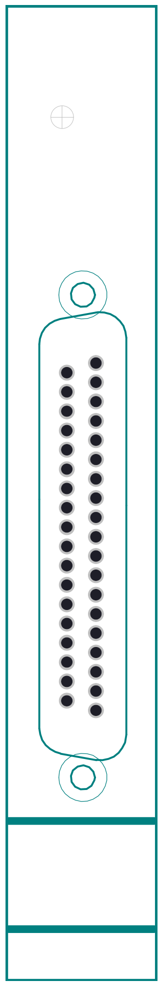


CCDs require bias voltages to power them, and scientific CCD cameras commonly require special circuits to power devices such as Thermoelectric Coolers (TECs), heaters and vacuum detectors, and to measure temperatures by means of thermistors. All of these functions require configurable analog voltages that are intimately related to the CCD and are capable of injecting electrical noise into the system.

In its standard configuration, the Service Module can provide a range of bias voltages sufficient to power all CCDs known to us. If more were required, or if multiple CCDs were to be controlled, then additional Service Modules may be used.

Power circuits are available for a broad range of heaters, TECs and vacuum detectors. In addition, digital detection circuits are included to measure up to three temperatures and the vacuum pressure indicated by a standard thermocouple vacuum detector.

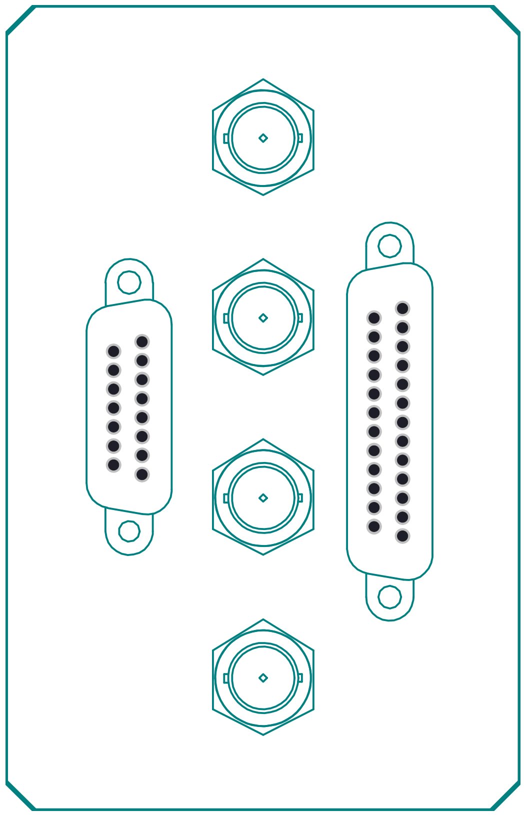
## Clock Driver Module



CCDs use clock voltages to move charge around either by means of parallel clocks in image arrays or by serial clocks in registers. The number of clocks and their voltage requirements vary from CCD to CCD, although generally 3 clock phases are used in image arrays and 3 clocks are used in serial registers. In the case of a frame storage CCD, 3 clocks are generally used for the image array, 3 for the frame store and 3 for the serial register. However, it is not uncommon to have 2, 4 or more clocks for any particular function. In addition, there is usually also a reset clock, which is used to conduct residual charge away from the output, in preparation for the charge of the next pixel. These clock voltages generally fall within the limits of the bias voltages used to power the CCD.

The Clock Driver Module is designed to drive all CCDs known to us with minimal modification through configuration. Up to 7 voltage pairs can be provided and these can be jumpered through a selection matrix into up to 15 clock output drivers. If a CCD were to require more clocks, or if multiple CCDs were to be controlled, then additional Clock Driver Modules can be used.

## Camera Head

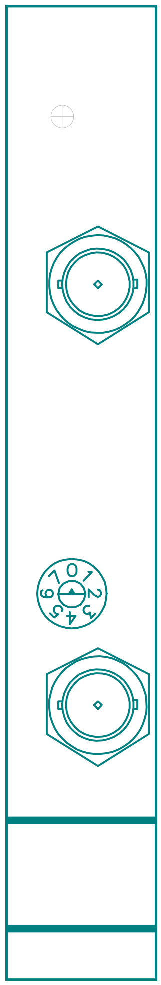


Generally, to obtain the very best read-noise performance, it is best to have the CCD and a Preamplifier Module in a Camera Head that is separated from the CCD Controller and connected by cables. This also makes it simple to control multiple CCDs from a single CCD Controller. Although the Preamplifier Module is not technically part of the Controller, and does not plug into the Controller Bus, it serves to make the CCD an abstract entity. It does this by conditioning the clocks and biases from the Clock Driver and Service Modules and processing the video outputs from the CCD to provide a standard responsivity and drive the video cables.

The CCD clocks must have fast rise times in order to traverse the cable without losing integrity. However, the fast rise times would also lead to high frequency harmonics in the camera head, and generate noise in the signal. The clocks must therefore be conditioned with filters on the Preamplifier module to achieve the appropriate rise times. Experience has indicated that an optimal value for responsivity from the CCD is 20 microvolts of signal for each electron of charge. Impedance-matched cable drivers send the video signal to the Controller over 75 ohm cables.

The camera head shown in this illustration is typical for a four-port CCD such as the Marconi CCD39, or the MIT/LL CCID series.

## Input Module



The Input Module receives the video signals from the Camera Head and converts them to digital data. Although most CCDs have a single video output, many scientific CCDs have multiple outputs in order to increase the maximum frame rate relative to the pixel rate, which is what determines the minimum read-noise. Input Modules may be designed to have one or more video channels and multiple Input Modules may be used for multiple output CCDs, or for multiple CCDs controlled by a single controller.

Scientific CCD cameras ideally exhibit the following properties. They must have a range of at least two gain settings such that the lowest gain allows the observation of the maximum CCD signal, and the highest gain allows resolution of individual electrons. The lowest gain varies by CCD and the highest gain varies by A/D converters, but is generally 0.5 electrons per Data Number. They must also run at a variety of frame and pixel rates and thus must have several selectable low pass filters in the video chain. Even more importantly, the video amplifier response must be linear and calibratable.

Like many CCD cameras, our CCD Controller utilizes Correlated Double Sampling when digitizing the video signal. This is done by capacitively coupling the video signal and clamping it to a clamp (or offset) voltage using a Clamp signal that is part of the sequence. The Clamp signal is held on until the video signal is pulled to the clamp voltage level. This defines the lower limit of the video signal that is digitized. Then the pixel charge is moved onto the output of the CCD and the voltage level will rise in proportion to the charge in the pixel. The A/D converter is made to convert this analog voltage into digital data by means of a Sample signal that is also part of the sequence. The Reset clock is then asserted and the whole cycle starts over again. The offset voltage is controlled through the $I^2C$ Bus by a 10 bit Digital to Analog (D/A) converter. Fine control of the offset voltage allows the maximization of the dynamic range of the signal and enables equalization of multiple video channels when necessary.

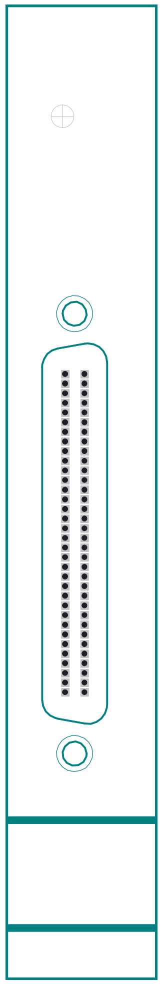
The timing and duration of Clamp and the timing of Sample relative to Reset are absolutely critical to achieving the lowest possible read-noise from the CCD. In this regard, the clock frequency of 50 MHz results in a relatively crude resolution of 20 nSec. Uniquely in this design, both the Clamp and Sample signals are run through digital delay lines that allow adjustment of their phases relative to Reset in increments of 0.25 nSec. The digital delay lines are controlled through the $I^2C$ Bus.

Gain and filter selections are made by controlling relays through the $I^2C$ Bus and their states can be stored in the Flash RAM in the Command Module. Similarly, the offset values and the clamp and sample delays can also be stored in the Flash RAM.

## Digitizing Submodules

In general, there is a trade off in A/D converters between speed and bit resolution. There is also a wide variety of A/D converters available at any given speed and bit resolution. These vary in many ways, ranging from voltage conversion range to pipeline depth in the digital readout to cost. In addition, A/D converters are in a class of semiconductors whose price and availability is highly variable due to their increasing use in consumer products and surges in their popularity. For all these reasons, it is best to abstract the A/D converter by means of incorporating it into an ADC Submodule, which is provided with a standard footprint on the Input Module. This also leads to an extremely short time to market for new A/D converters.

## Output Module



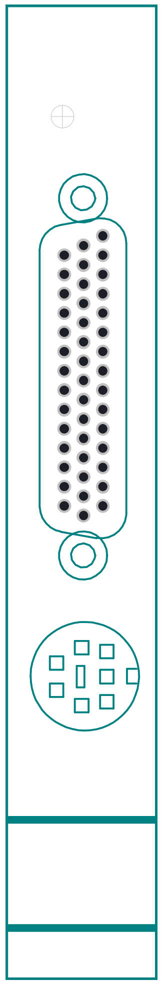
A variety of digital camera formats exist that can easily be derived from the SOF (Start of Frame), SOL (Start of Line), DATA READY special Control codes and the Image Data. All of the information required is available as it is generated from the CCD. In particular, two forms of digital camera interface have been implemented, as follows.

The first is in a demultiplexed form where the digital data from each channel is presented at a unique connector and overflow and handshaking information (SOF, SOL, etc) is available at a separate connector. This is the simplest, most efficient and most effective means of transmitting the data for real-time processing.

The second form (shown at left) uses the standard AIA protocol for digital cameras. This standard is widely used and there is a vast array of products available for a variety of computer platforms and operating systems that support the standard. In the case of multiple output CCDs and in controlling multiple CCDs from a single controller, it is particularly useful to have all the data integrated into a single data stream for the purposes of time registration and data manipulation. The AIA standard also incorporates an RS-232 serial port and specifies a command protocol, which we have implemented.

A variety of output modes may be required at any one time, and for this purpose, multiple Output Modules may be used simultaneously.

## Timing Module



The Timing Module is a special purpose device and will not be found in all cameras. Its primary purpose is to synchronize the operation of up to six otherwise independent Little Joe cameras. Like other modules, it resides on the Controller Bus and communicates with a Command Module via the $I^2C$ protocol.

The six outputs of the Timing module connect to the RSI inputs of the Command Modules of other cameras. Each camera's frame rate can then be set individually, without requiring separate communications links to all of them.

The Timing Module also has an RSI of its own, so that the whole multi-camera system may be synchronized with an external device.

It can also drive the master clock signal on the Controller bus if a resolution other than (less than) 50 MHz is required (such as for compatibility with other systems).

## Backplane

The fundamental property of the design is the CCD Controller Bus. The bus incorporates all the features required to control all known CCDs. These include a Digital Power Bus, an Analog Power Bus, an $I^2C$ Serial Bus, an RS-232 Serial Bus, a Digital Sequence Bus and an Image Data Bus.

Although the Controller Bus can take a variety of physical forms, we have elected to use circuit boards that conform to a standard 3U form factor. Specifically, the slot spacing is 0.8" (or 4 HP) and industry standard enclosures such as those used for VME bus use 7, 10, 15 and 21 slots.

The Digital Power Bus provides +5V and Digital Ground to all the slots to provide power to the digital circuits within the camera. Digital circuits are notoriously noisy and it is important to isolate the analog circuits from digital noise to minimize read-noise.

The Analog Power Bus provides +12V, -12V, +5V, -5V, +24V and Analog Ground to all the slots to provide power to the analog circuits within the camera. All other analog voltages required can be derived.

The $I^2C$ Serial Bus is used to communicate between modules on the bus at relatively low speeds and relatively infrequently. This would be for purposes such as selecting a gain setting or reading a temperature and would not necessarily be synchronous with reading the image.

The RS-232 Serial Bus is for the purposes of external control. This is an important feature for the purpose of making the controller computer-controllable, but platform-independent. The RS-232 Serial Bus is on the backplane because the Command Module hosts the serial bus controller, but connection can be made through any module.

The Digital Sequence Bus is used to control all high speed events in the camera. These are typically clocks that control the CCD clock inputs, dedicated signals such as Clamp and Sample, and special Control bits. The clocks are obviously used to move the charge around on the CCD. Clamp and Sample are used in digitizing the video output from the CCD, and are used to implement Correlated Double Sampling. The special Control bits are used to control the flow of, and indicate the meaning of, the image data generated within the camera.

The Image Data Bus is used to propagate the image data as it is generated. The Control bits indicate the meaning of the image data.

THE "LITTLE JOE" DIGITAL CAMERA

# Users' Guide


1123 Zonolite Road • Suite 28
Phone 404-876-6558 • Fax 404-876-6004
Web www.SciMeasure.com

Chapter

1

# The Hardware Interface

*This chapter discusses the hardware requirements and low-level communications protocols.*

The Little Joe camera typically provides four interface connections. Two are redundant serial command ports, one is for synchronizing the camera to a data recorder, and the fourth is the data output stream. This chapter should help the user decide which serial port to use, how and whether to use the synchronizing interface, and how to receive streamed image data for processing and analysis.

## *The Serial Command Ports*

There are two serial ports on the camera for sending commands and receiving status information. The preferred connection is via the primary AIA connector on the Output Module (OM). This port follows the EIA-422A standard. Typically this connects to a frame grabber or other data input card that provides the 422 port. Communicating through this port then depends on the drivers supplied with the interface card.

The other connection is provided by the DB-9 port on the Command Module (CM). This is a basic RS-232C port, transmitting data on pin 2 and receiving on pin 3 (as a DCE); pin 5 is signal ground. This interface was designed to be used with an ordinary terminal or emulator. Pin 9 is held at +12 volts, and on some emulators this can cause connection troubles. The solution is to not connect pin 9 on the serial cable.

The OM port can be used if the computer that processes the image has sufficient resources to simultaneously run a camera controlling application. It also requires that the camera and computer be within the reach of the data cable, which is typically ten feet long.

The CM port would be preferred if the controller has to be at some distance from the camera, or if the image processing computer has to be dedicated to the task for technical reasons.

Either of these ports may be used, depending on the application, but due to electrical conflicts, only one port may be enabled at a time. To use the Command Module port, close switch SW1-RXD on the CM, and remove jumpers J3 and J4 on the OM. (If using revision "B" of the OM, open both switches on SW1.) To use the Output Module port, open the

SW1-RXD switch on the CM, and install the jumpers J3 and J4 on the OM. (If using revision "B" of the OM, close both switches on SW1.) If there is more than one OM in the system, only one should be enabled for communication – the others should have their jumpers (or switches) opened.

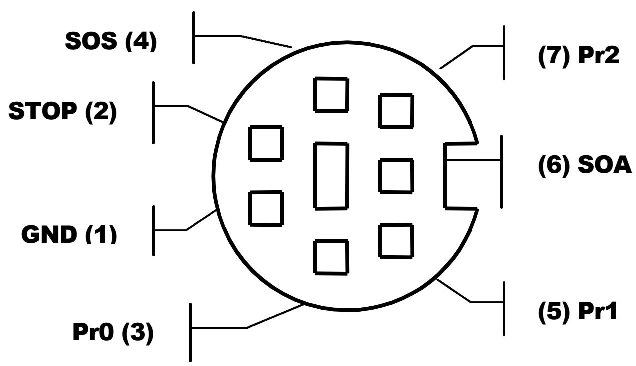
**Warning: turn off power to the camera before removing any modules. Further, if both ports are enabled at the same time and the camera is powered, damage to the modules will likely result.**

The camera does not provide character echo; use local echo if you need it. The standard connection is 8 bits, no parity, 1 stop bit. The communication protocol complies with AIA document BSR/AIA A15.08/7-199X (q.v.).

## *The Remote Synchronizing Interface*

In addition to the two communications ports, there is what is known as the Remote Synchronizing Interface (RSI) on the front panel of the Control Module. This interface is also discussed to some extent in the Command Reference under the "TXC" command. The pinout of the connector is shown in figure 1, below.

**FIGURE 1.** *A view of the connector as it appears looking toward the front panel. The signal names shown do not appear on the panel, but are shown for reference in the text.*




| Pin | Name | Function |
|---|---|---|
| 1 | Gnd | Ground; signal reference. |
| 2 | Stop | Bring to ground to run; float high (TTL or +5v) to stop. |
| 3 | Pr0 | Pr2, Pr1, and Pr0 are binary code inputs for the program number. |
| 4 | SOS | Start of Sequence; an output pulse. |
| 5 | Pr1 | (See Pr0) |
| 6 | SOA | Start of Accumulation; an output pulse. |
| 7 | Pr2 | (See Pr0) |

This interface allows frame rate control and program selection without use of the serial port. Although the functionality is limited, it provides a greater degree of precision than the serial port. Frames may be synchronized to another device, and the current program may be selected or changed synchronously for more complex observations.

### Signals

All signals are referenced to GND (ground), are TTL compatible, and active high. Input signals should not be driven, but allowed to float high, using the internal 4.7K pull-up resistors.

SOA and SOS are outputs, Pr0, Pr1, Pr2, and STOP are inputs. Although the output signals are always present, the system will not respond to the inputs unless it has been commanded to do so via the serial port. Refer to the "TXC" command in the Command Reference section of this document. Program selection is via Pr0, Pr1, and Pr2, with Pr0 being the LSB. A special condition exists when all three of these lines are high, as they would be when nothing is connected to the RSI. In this case, program selection reverts to the serial port, and whatever program was last selected there, becomes the operative one.

The behavior of the system in response to STOP varies, depending on the external mode selected. This is expanded below, under "Operation." Simply put, when STOP is high, the frame readout will stop at the end of the current frame. When STOP is brought low (i.e., to GND), it will recommence. This is how synchronization is accomplished.

SOS is the Start of Sequence signal. It is a short (20 nSec) pulse that occurs at the beginning of each frame. It can be used to count frames, or as a confirmation that the STOP line has been asserted.

SOA is the Start of Accumulation signal. Its operation is useful mainly when very long exposures (minutes or hours) are required. It is a short (20 nSec) pulse that occurs once per frame, *plus* the number of accumulator repetitions selected via the serial control (as with the "REP" command).

### Operation

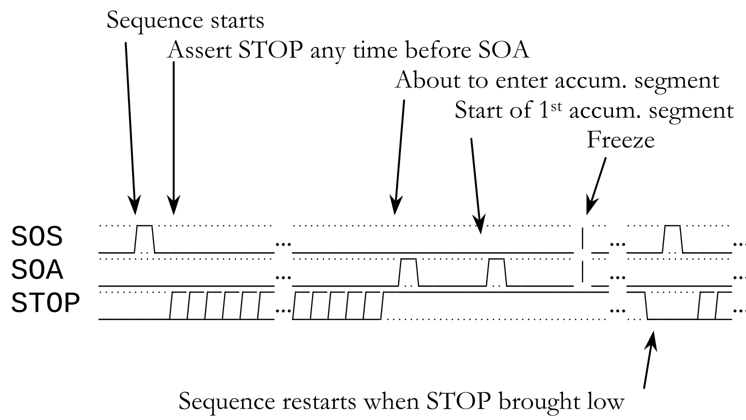
Normally, Little Joe will ignore activity at the RSI. It must be set running, then put into one of the two external modes before the RSI will be recognized. The modes are referred to as Freeze and Repeat.

To understand the difference between the two modes, an explanation of sequence construction is necessary. A sequence consists of several stages or segments that occur in order. These segments control such operations as generating frame enable, line enable, pixel read, and so on. The last segment in any sequence is the accumulation (or integration) segment. This may be a short do-nothing operation, or may involve dithering on long exposures to reduce dark current. Conceptually this segment should occur at the beginning of a frame, because it is here that the light is collected for the readout which follows. However, certain design constraints have caused it to be placed at the end. This slight conceptual anomaly is of no consequence when frames are being continuously collected in a cyclical manner, but may become more important in more complex sequences.

In any case, the number of times this accumulation segment is repeated each frame is determined by the repetitions command ("REP"), but is always greater than zero.

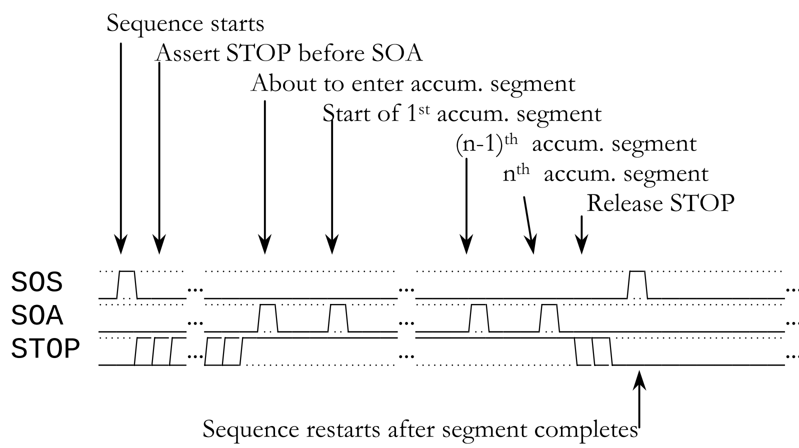
In Freeze mode (figure 2), the frame starts with a pulse on SOS. Once this pulse has occurred, the STOP line may be brought high at any time before the first SOA pulse happens. The time lag between these events is determined by the intrinsic frame rate of the selected program. When STOP is high, the sequence *will continue to run* to the end, including all selected accumulator repetitions, at which point it will stop (freeze). The readout of the CCD is complete, but it continues to gather light. When STOP is brought low, the process starts over.

**FIGURE 2.** *(not to scale) The general relationships among SOS, SOA and STOP in Freeze mode. In this example, the number of accumulation repetitions is set to 1.*




Repeat mode (figure 3) is the same as Freeze mode, except that once the sequence reaches the end, it does not freeze; rather it repeats the accumulation segment continually until STOP is de-asserted. It will then run to the end of the accumulation segment in progress, then start the sequence anew.

**FIGURE 3.** *(not to scale) The general relationships among SOS, SOA and STOP in Repeat mode. If multiple repeats are programmed, they will run to completion before the de-assertion of STOP is recognized.*



## *Image Data Interface: Streamed*

Little Joe's Output Module provides the means to stream data out to the image processing computer. The connector and signals follow applicable industry standards, and are compliant with AIA specification A15.08/2-1993 Rev. 8, specifically the parts that pertain to extended single-channel digital cameras.

The connector is a 68 pin microminiature D type commonly used in SCSI-III applications. Signals are differential, and follow the EIA-422A standard.

| Pin | Signal | Source |
|---|---|---|
| 1 | Ground | |
| 2 | MSB (+) | Camera |
| 3 | MSB-1 (+) | Camera |
| 4 | MSB-2 (+) | Camera |
| 5 | MSB-3 (+) | Camera |
| 6 | MSB-4 (+) | Camera |
| 7 | MSB-5 (+) | Camera |
| 8 | MSB-6 (+) | Camera |
| 9 | MSB-7 (+) | Camera |
| 10 | MSB-8 (+) | Camera |
| 11 | MSB-9 (+) | Camera |
| 12 | Ground | |
| 13 | MSB-10 (+) | Camera |
| 14 | MSB-11 (+) | Camera |
| 15 | MSB-12 (+) | Camera |
| 16 | MSB-13 (+) | Camera |
| 17 | No connect | |
| 18 | No connect | |
| 19 | MSB-14 (+) | Camera |
| 20 | MSB-15 (+) | Camera |
| 21 | Reserved | |
| 22 | Serial Out (+) | Camera |
| 23 | Serial In (+) | Processor |
| 24 | Field ID (+) | Camera |
| 25 | Frame Enable | Camera |
| 26 | Line Enable (+) | Camera |
| 27 | Channel ID0 (+) | Camera |
| 28 | Channel ID1 (+) | Camera |
| 29 | Pixel Strobe (+) | Camera |
| 30 | Mode 0 (+) | Processor |
| 31 | Mode 1 (+) | Processor |
| 32 | Mode 2 (+) | Processor |
| 33 | Mode 3 (+) | Processor |
| 34 | Ground | |

| Pin | Signal | Source |
|---|---|---|
| 35 | Ground | |
| 36 | MSB (-) | Camera |
| 37 | MSB-1 (-) | Camera |
| 38 | MSB-2 (-) | Camera |
| 39 | MSB-3 (-) | Camera |
| 40 | MSB-4 (-) | Camera |
| 41 | MSB-5 (-) | Camera |
| 42 | MSB-6 (-) | Camera |
| 43 | MSB-7 (-) | Camera |
| 44 | MSB-8 (-) | Camera |
| 45 | MSB-9 (-) | Camera |
| 46 | Ground | |
| 47 | MSB-10 (-) | Camera |
| 48 | MSB-11 (-) | Camera |
| 49 | MSB-12 (-) | Camera |
| 50 | MSB-13 (-) | Camera |
| 51 | No connect | |
| 52 | No connect | |
| 53 | MSB-14 (-) | Camera |
| 54 | MSB-15 (-) | Camera |
| 55 | Reserved | |
| 56 | Serial Out (-) | Camera |
| 57 | Serial In (-) | Processor |
| 58 | Field ID (-) | Camera |
| 59 | Frame Enable (-) | Camera |
| 60 | Line Enable (-) | Camera |
| 61 | Channel ID0 (-) | Camera |
| 62 | Channel ID1 (-) | Camera |
| 63 | Pixel Strobe (-) | Camera |
| 64 | Mode 0 (-) | Processor |
| 65 | Mode 1 (-) | Processor |
| 66 | Mode 2 (-) | Processor |
| 67 | Mode 3 (-) | Processor |
| 68 | Ground | |

Little Joe at this time does not make use of the Field ID, Channel ID or Mode signals.

## *Image Data Interface: Demultiplexed*

An alternative method of acquiring image data can be provided by an Output Module that is configured to present demultiplexed data to the processing computer. This method is advantageous for multiple-channel CCDs, because all channels can be processed simultaneously. (With a streamed output, all channels have to be finished before a frame can be presented.)

Physical and mechanical restraints prevent demultiplexed outputs from appearing on a front panel connector. Instead, there is a set of vertical headers that interface to regular IDC ribbon cable connectors. The ribbon cables may then be brought out wherever there is clearance for them.

The Output Module is available in two revisions for this purpose. Revision "A" provides four, twelve-bit outputs, and revision "B" provides four, sixteen-bit outputs. When using an ADC of other than 12 or 16 bits, it is necessary to shift the data (using the "MAP" command) to properly register the data.

In addition to the four data outputs, another connector provides the control signals Start of Frame (SOF), Start of Line (SOL), and Data Ready (DRDY). These signals are true low, and have the following timing relationship:

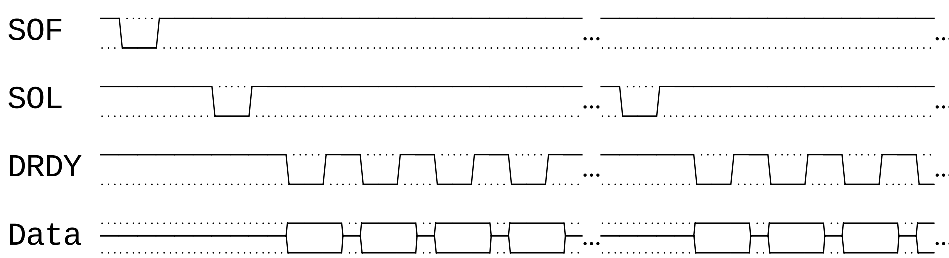



**FIGURE 4.** *(not to scale) The timing relationship for the demultiplexing outputs.*

On revision "A" these signals are on P5; on "B" they are on P3. In either case, they have the pinout shown in **Table 1**, below.

The signals OV1 – OV4 are the overflow bits for their respective channels. Note that not all A/D Converters supply an overflow bit, so this signal may not be present on any given system. Process This Pixel (PPIX) is always true (high level) in this implementation.

The data channel connector pinouts are shown in **Table 2** for the revision "A" modules, and in **Table 3** for revision "B."

Following the tables are sketches of the modules, showing the placement and orientation of the demultiplexing connectors. Recall that not all Output Modules have these connectors or the demultiplexing capability.

| Revision "A", P5<br>Revision "B", P3 | |
|---|---|
| **Pin** | **Function** |
| 2 | OV1 |
| 4 | OV2 |
| 6 | OV3 |
| 8 | OV4 |
| 10 | PPIX |
| 12 | DRDY |
| 14 | SOF |
| 16 | SOL |
| 18 | N/C |
| 20 | N/C |
| Odd | Ground |

**Table 1.** *Demultiplexing control signals.*

| Revision "A", P1 – P4 | |
|---|---|
| **Pin** | **Function** |
| 2 | D0 |
| 4 | D1 |
| 6 | D2 |
| 8 | D3 |
| 10 | D4 |
| 12 | D5 |
| 14 | D6 |
| 18 | D7 |
| 20 | D8 |
| 22 | D9 |
| 24 | D10 |
| 26 | D11 |
| Odd | Ground |

**Table 2.** *Data output connections for revision ""A."*

| Revision "B", P1 – P2 | |
|---|---|
| **Pin** | **Function, Channel 1 (P1) or 2 (P2)** |
| 1 | D0 |
| 3 | D1 |
| 5 | D2 |
| 7 | D3 |
| 9 | D4 |
| 11 | D5 |
| 13 | D6 |
| 15 | D7 |
| 17 | D8 |
| 19 | D9 |
| 21 | D10 |
| 23 | D11 |
| 25 | D12 |
| 27 | D13 |
| 29 | D14 |
| 31 | D15 |
| **Pin** | **Function, Channel 3 (P1) or 4 (P2)** |
| 33 | D0 |
| 35 | D1 |
| 37 | D2 |
| 39 | D3 |
| 41 | D4 |
| 43 | D5 |
| 45 | D6 |
| 47 | D7 |
| 49 | D8 |
| 51 | D9 |
| 53 | D10 |
| 55 | D11 |
| 57 | D12 |
| 59 | D13 |
| 61 | D14 |
| 63 | D15 |
| Even | Ground |

**Table 3.** *Data output connections for revision "B."*

**FIGURE 5.**

*Revision "A" demultiplexing connections.*

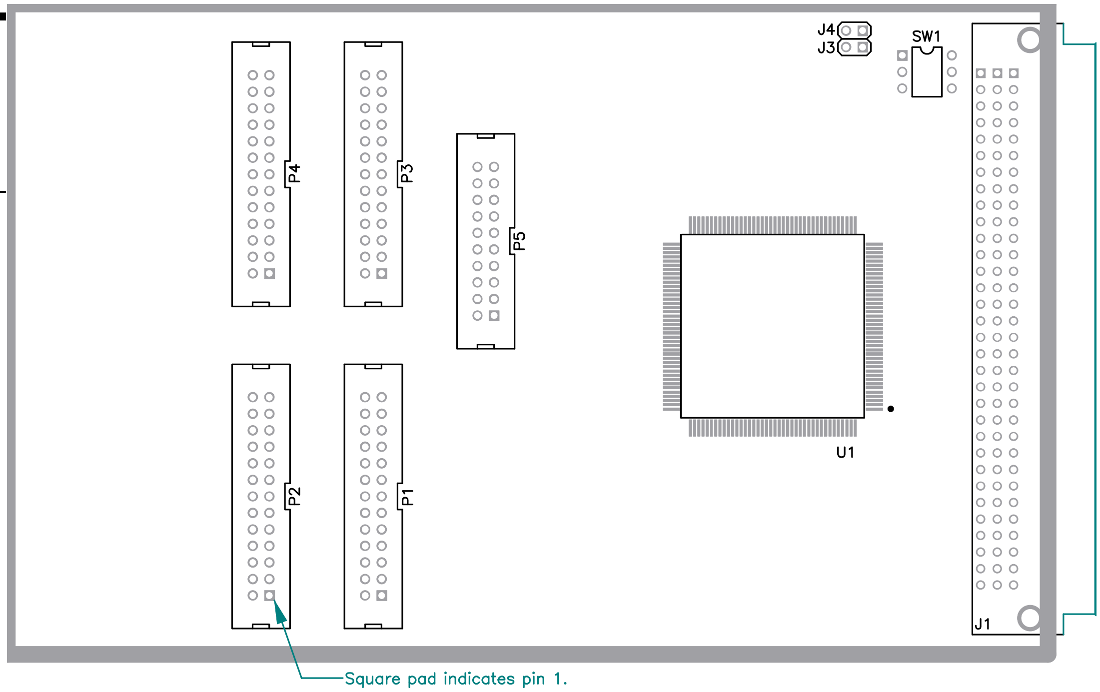
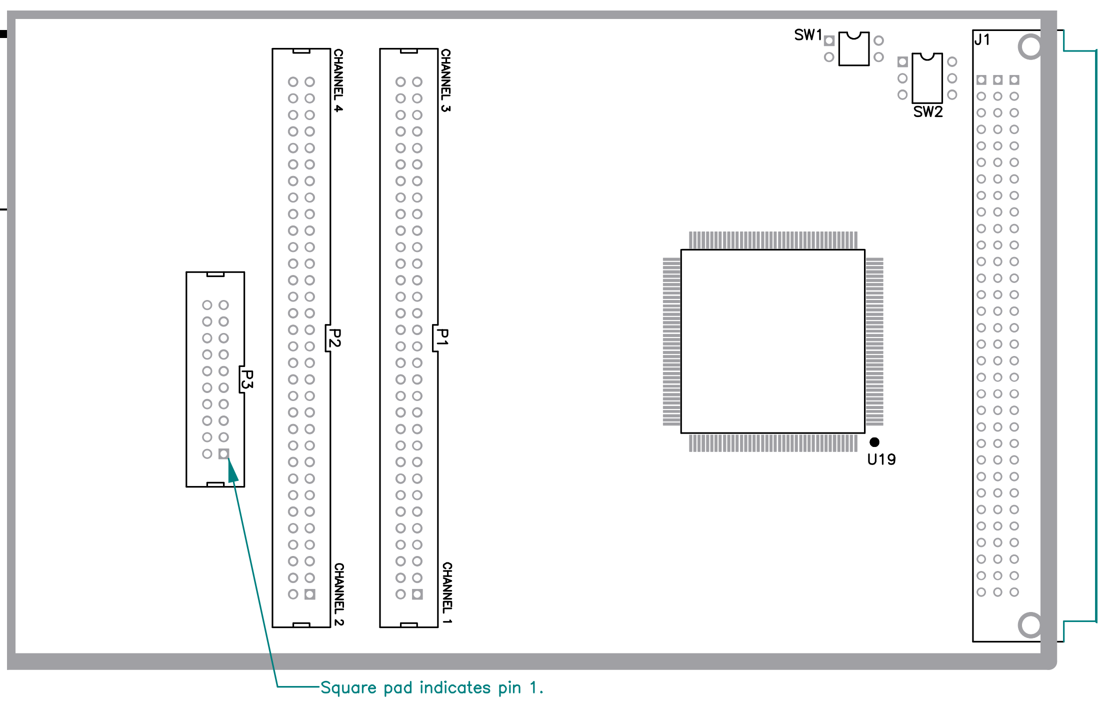



**FIGURE 6.**

*Revision "B" demultiplexing connections.*



Chapter

2

# The Command Set

*This chapter describes the three main types of commands, lists each command and gives the syntax for each.*

In compliance with the AIA specification, all Little Joe commands are three upper case characters (trigraph) preceded by "@". A question mark after the command (no whitespace) indicates a query sent to the camera. An exclamation point indicates that it is the camera's reply to a query. In most cases, whitespace can be added for readability. The termination of the command is indicated by <CR> which, being unprintable, is not shown in the syntax.

Little Joe does not provide character echo; the controlling application should provide it if needed.

Numerical input parameters can be supplied either as plain decimal, hexadecimal if prefixed with "$" (e.g. $FF), or binary if prefixed with "&" (e.g. &0010). When text is a parameter, it is prefixed with a single quote ('), and limited to the printable ASCII character set.

Little Joe's receive queue is unbuffered. It sends an ACK (06 hex) each time a command is fully processed. Subsequent commands sent before the ACK is received risk overwriting the previous command. (Exceptions: XMC and XMP, where ACK would confuse the Xmodem handshake.)

In general, Little Joe does not volunteer or "push" information. Messages will be sent from the camera only in response to queries. The principal exception is at bootup, when the camera sends a few startup status messages, ending with "Little Joe x.x.x is ready." The "x.x.x" represents the software version number. Other exceptions are noted in the reference that follows.

Little Joe's command set may be grouped into three categories:

1. Those that control camera operation for or during the production of images,
2. Those that manage the files that make the camera operate, and
3. Other miscellaneous commands.

### *Most Frequently Used Camera Operation Commands*

Little Joe can be fully operated with just a few commands, although many are provided. Most of the commands in this reference are low-level instructions more useful to developers than to end-users. Very few users will ever need the file management or miscellaneous commands. The most useful ones are those that control camera operation. The commands most frequently used are:

- Recall Settings (RCL) lets the user select one of eight pre-programmed operational modes, or "personalities." Typically, you would use this to select the frame rate, at the same time recalling all the appropriate filtering, offsets, and other parameters that are optimized for that speed.
- Start / Stop (SEQ) controls image and data acquisition.
- Offset (OAC, OIC) is used together with "Attenuation" to obtain the best image for the specimen or object being viewed.
- Attenuation (AAM) also known as "Gain" is used to alter image intensity.
- Temperature (TMP) It is useful to monitor the temperature periodically for two reasons. The image data will not be reliable if the CCD temperature is not stable; secondly, system failures may be avoided if the system is not allowed to overheat.

The reference that follows is in alphabetical order, by command trigraph. The top of each page indicates whether the command is considered pertaining to Camera Operation, File Management, or Miscellaneous usage.

## Attenuation

There are two commands that pertain to signal attenuation (also referred to as gain or amplification). The attenuation of each Input Module (but not each channel) can be controlled individually, or for convenience, all together.

**AAM** Attenuate All Modules

**AIM** Attenuate Individual Module

**Command syntax:**

`@AAM v`

Set all Input Modules to the same attenuation value *v*.

`@AIM #m:v`

Set Input Module *m* to attenuation value *v*

**Query syntax:**

`@AAM?`

Query the attenuation values of all Input Modules.

`@AIM? #m`

Query the attenuation value of Input Module *m*.

**Reply format:**

`@AAM! #0:v; #1:v; … #n:v`

List the attenuation values of Input Modules 0 – *n* (as many as are present in the system, up to 8). Each module will respond individually, even if they all have the same attenuation.

`@AIM! #m:v`

The attenuation value *v* of Input Module *m* is given in response to the query.

**Errors:**

AAM will return error 5 if *v* is out of range. AIM will return error 5 if *v* is out of range, or if *m* is greater than or equal to the number of Input Modules present. The Input Module number denoted by *m* can be read on the dial switch on the face plate of the module.

**Discussion:**

The value *v* is simply an index number from 0 (zero) to 3, with 0 representing the least attenuation. The amount of attenuation indicated by values 1, 2, and 3 can vary depending on the hardware configuration, but the standard values are:

| Value | Attenuation (dB) | Gain (dB) | Gain ($e^-$/DN) | Amplification (V/V) |
|---|---|---|---|---|
| 0 | 0 | +24 | 0.4 | 16.5 |
| 1 | 10 | +13 | 1.4 | 4.50 |
| 2 | 20 | +3.5 | 4.0 | 1.50 |
| 3 | 30 | -7.7 | 14 | 0.41 |

Note that while attenuation or amplification can be controlled only in the Input Modules, there are other gain stages in the system that are taken into account to derive the figures shown above.

**Examples:**

`@AAM 3`

Set all Input Modules to maximum attenuation.

`@AIM #1:0`

Set Input Module "1" to minimum attenuation.

`@AAM?`

`@AAM! #0:1; #1:1`

A query, and a response that indicates there are two input modules, both with an attenuation value of 1 (low).

`@AIM? #2`

`@ERR^5`

A query, and a response that indicates that Input Module "2" (the third module) is not present or detected.

# Baud Rate

The communications link baud rate is accessible through this command.

**BAU** Baud rate

**Command syntax:**

@BAU *v*

Set the baud rate to *v*, which may have any of the values 600, 1200, 2400, 4800, 9600, 19200, 38400, or for short, 6, 12, 24, 48, 96, 192, or 384.

**Query syntax:**

@BAU?

Query the baud rate.

**Reply format:**

@BAU! *v*

The baud rate is returned in decimal form, in response to a query.

**Errors:**

BAU returns error 3 if an invalid value is given.

**Discussion:**

The baud rate applies to both the Command Module port and the Output Module port, so changing it in one place changes both. The value set by this command is *not* retained after a shutdown, unless the SAV 0 command is invoked. Parameter set 0 is the one loaded at bootup, and is the only one that can set the default baud rate.

To change the baud rate in a meaningful way, obviously both ends of the connection must be changed. The default baud rate is the fastest one available on the Little Joe. If a lower rate is needed because of environmental conditions, it can be achieved. A series of events like this would have to occur:

1. Connect at the default baud rate (using a short cable, in a "clean" environment).

2. Issue the "@RCL 0" command, so that when the new baud rate is saved later, it will save all the other settings correctly.

3. Issue the BAU command (e.g. @BAU 9600).

4. Change the baud rate at the controller (e.g. with pdv_set_baud() ).

5. The rates should now match. Test the link with "@BAU?" or any other query.

6. Issue the "@SAV 0" command to save the new rate as the default.

7. It should now be safe to set up the equipment in the less optimal conditions.

**Examples:**

```
@BAU 96
@BAU 9600
```

Either of these will set the baud rate to 9600.

```
@BAU?
@BAU!38400
```

The baud rate is queried, and shown to be 38400.

## File Saving

Sequences loaded into RAM can be transferred to non-volatile Flash memory with these two commands.

**CTF** Control To Flash

**PTF** Pattern To Flash

**Command syntax:**

`@CTF 'control file description`

`@PTF 'pattern file description`

Issuing either of these commands causes the corresponding RAM file to be copied to Flash memory. The previous contents of Flash, if any, are destroyed, but the RAM is not.

The file name parameters are optional and need not be supplied. Their only purpose is to be returned when they are queried, as a mnemonic help to the user. They can consist of up to 56 characters from the ASCII standard printable set ("space" through "tilde"). Any other character automatically terminates the string.

The file names must be delimited at the beginning by a single quote (ASCII character 27 hex).

**Query syntax:**

`@CTF?`

`@PTF?`

The queries can be used to retrieve the file names, if any, supplied when the files were last copied to Flash.

**Reply format:**

`@CTF!`

`@PTF!`

These replies are returned automatically from the CTF and PTF commands, when the files are successfully copied. The copy operation can take up to 60 seconds, and these replies are signals that the operation is complete. No command, queries, or other communications should be attempted between the CTF/PTF command and the corresponding response.

This is one of the few exceptions to the "no pushing of information" principle. It is necessary here, because the system will otherwise appear hung up or unresponsive.

`@CTF! 'control file description`

`@PTF! 'pattern file description`

These replies are given in response to the corresponding queries. If file names were saved, they are returned here.

**Errors:**
Error code 4 is returned if any character other than white space occurs after the text of the command, before the single quote.

**Discussion:**
These commands are useful only to the user who writes his own sequences. Trial sequences, once compiled (or supplied from a vendor) can be uploaded to the camera via the File Transfer commands XMC and XMP (q.v.). If the user decides to keep them instead of the sequences already in Flash, these commands are used to accomplish that.

Since the AIA protocol does not support the single quote or text in the data field, these commands are guaranteed to be available only through the Command Module port. If a third-party system is driving the camera through the Output Module, these commands might not be available.

**Examples:**

`@XMC`

(… the terminal program then initiates Xmodem/CRC transfer of a file, e.g. "shempfav.bin" …)

(… several seconds elapse, during which the file is transferred …)

`@XMO! $000064`

(… signals the end of the transfer. The user may then send …)

`@CTF 'Shemp Howard's Favorite Control File`

(… several seconds elapse …)

`@CTF!`

(… signals the restoration of communications. The user may then query …)

`@CTF?`

`@CTF! 'Shemp Howard's Favorite Control File`

This series of commands and responses shows how a Control file, of about 12800 bytes, is transferred and saved.

# Debug

An onboard debugger can be started.

**DBG** Debug

**Command syntax:**

`@DBG`

Starts the debugger.

**Query syntax:**

There is no query or reply allowed for this command.

**Errors:**

There is no error returned for this command.

**Discussion:**

This is a powerful and destructive command, and its use is not recommended. The user should be aware of its existence so as not to invoke it by accident. It is possible, though unlikely, to damage the camera by injudicious use of the debugger. It is very easy, however, to damage the contents of the Flash memory upon which proper operation depends.

If the debugger should be launched inadvertently, the user will be presented with the prompt:

```
debug>
```

There may be a menu of commands presented as well, but the debug prompt will always show up at the end of it. If you see this prompt, just type the letter x or X to exit the debugger. It would probably be a good idea to reboot the camera.

**Examples:**

```
@DBG
debug>x
```

The debugger was accidentally launched and safely exited.

## Signal Delays

There are four commands that pertain to adjusting or fine-tuning the timing of the sample and clamp signals. Each Input module may be controlled independently, or for convenience, all may be controlled together.

**DCA** Delay Clamp All input modules

**DSA** Delay Sample All input modules

**DCI** Delay Clamp Individual module

**DSI** Delay Sample Individual module

**Command syntax:**

`@DCA` *v*

Set all Input Modules to delay the clamp signal by *v* x 0.25 nSec.

`@DSA` *v*

Set all Input Modules to delay the sample signal by *v* x 0.25 nSec.

`@DCI #`*m*`:`*v*

Set Input Module *m* to delay the clamp signal by *v* x 0.25 nSec.

`@DSI #`*m*`:`*v*

Set Input Module *m* to delay the sample signal by *v* x 0.25 nSec.

**Query syntax:**

`@DCA?`

Request a list of the clamp delay settings for all present Input Modules.

`@DSA?`

Request a list of the sample delay settings for all present Input Modules.

`@DCI? #`*m*

Request the clamp delay setting for Input Module *m*.

`@DSI? #`*m*

Request the sample delay setting for Input Module *m*.

**Reply format:**

`@DCA! #0:`*v*`; #1:`*v*`; … #`*n*`:`*v*

`@DSA! #0:`*v*`; #1:`*v*`; … #`*n*`:`*v*

List the clamp (DCA) or sample (DSA) values of Input Modules 0 – *n* (as many as are present in the system, up to 8). Each module will respond individually, even if they all have the same setting.

`@DCI! #`*m*`:`*v*

`@DSI! #`*m*`:`*v*

The clamp (DCI) or sample (DSI) value *v* of Input Module *m* is given in response to the query.

**Errors:**

Error 5 will be returned for DCI and DSI if *m* is out of range for the set of Input Modules present. DCI and DSI ought to have error checking, but as of version 2.2.0, they do not. Values greater than 255 will "wrap around" and yield modulo 256. Users should not depend upon this behavior, as it is expected that some future version will have range checking, to return the standard error 5.

**Discussion:**

These commands are generally not useful, except for users who write their own CCD sequences or programs. The signal delays are optimized at the factory for the programs shipped with the camera.

The timing of the sample and clamp signals is crucial to getting the best results from the camera. The base timing is set in the sequence, then the signals are delayed in increments of 0.25 nSec, up to a total of 64 nSec. Acceptable values for *v* are therefore 0 – 255.

**Examples:**

`@DCA 80`

Set the system clamp delay to 20 nSec later than that called for by the controlling sequence.

`@DSI #1:50`

Set the sample delay of Input Module 1 to 12.5 nSec.

# Erase

Sequence files and operational settings can be permanently discarded with this command.

**ERA** Erase

**Command syntax:**

`@ERA v`

If *v* is 0 (zero), all the camera operational settings (see SAV and RCL) are erased. The current operational settings are unaffected.

If *v* is 1 (one), the sequence files (both Control and Pattern) saved in Flash memory are erased. Files currently in sequencer RAM are unaffected.

**Query syntax:**

There is no query or reply allowed for these commands.

**Errors:**

Error 5 is returned if any parameter other than 0 or 1 is sent.

**Discussion:**

This is a powerful and destructive command, and its use is not recommended. The user should be aware of its existence so as not to invoke it by accident.

Note that ERA 0 loses the default baud rate. If power is lost before SAV 0 can be invoked, the camera will attempt to auto-detect the baud rate at the next bootup, upon receipt of a *single* character at either port.

**Examples:**

`@ERA 0`

Discard the contents of all eight parameter storage registers.

`@ERA 1`

Discard all the stored sequencer files.

# Error

As a command, it only serves to test the communications link. It will also be seen as a response to an erroneous message.

**ERR** Error

**Command syntax:**

`@ERR`

Issues the Error test command.

**Query syntax:**

`@ERR?`

This syntax also issues the Error test command.

**Reply format:**

`@ERR^0`

This is the only reply to be expected from either the query or the command.

`@ERR^`*v*

The value *v* is an error code sent in response to another command.

**Errors:**

| | |
|---|---|
| 0 | No error |
| 1 | Parity error |
| 2 | Unrecognized command |
| 3 | Value format error |
| 4 | Unrecognized character |
| 5 | Value out of range |
| 6 | Checksum error |
| 100-199 | Xmodem errors |
| 200-299 | $I^2C$ errors |

**Discussion:**

The AIA specification requires this command, to test and verify the camera's error messaging. It may be used to ping the camera.

There is another ping command in the Little Joe, not specified by the AIA, and which does not exercise the error handler. If the ASCII code DLE (Ctrl-P or 10 hex) is sent, a lower case "p" is returned as a ping.

**Examples:**

`@ERR`

`@ERR^0`

The command and its reply.

## Filter

There are two commands that pertain to filtering the signal frequency response. The response of each Input Module (but not each channel) can be controlled individually, or for convenience, all together.

**FAM** Filter All Modules

**FIM** Filter Individual Module

**Command syntax:**

`@FAM v`

Set all Input Modules to the same filter value *v*.

`@FIM #m:v`

Set Input Module *m* to filter value *v*.

**Query syntax:**

`@FAM?`

Query the filter values of all Input Modules.

`@FIM? #m`

Query the filter value of Input Module *m*.

**Reply format:**

`@FAM! #0:v; #1:v; … #n:v`

List the filtration values of Input Modules 0 – *n* (as many as are present in the system, up to 8). Each module will respond individually, even if they all have the same filtering.

`@FIM! #m:v`

The filter value *v* of Input Module *m* is given in response to the query.

**Errors:**

FAM will return error 5 if *v* is out of range. FIM will return error 5 if *v* is out of range, or if *m* is greater than or equal to the number of Input Modules present. The Input Module number denoted by *m* can be read on the dial switch on the face plate of the module.

**Discussion:**

The value *v* is simply an index number from 0 (zero) to 3, with 0 representing the least filtration. The amount of filtration indicated by values 1, 2, and 3 can vary depending on the hardware configuration, but the standard values are:

| Value | Corner Frequency | Time Constant (nS) |
|---|---|---|
| 0 | 15.9 MHz | 10.0 |
| 1 | 5.64 MHz | 28.2 |
| 2 | 194 kHz | 820 |
| 3 | 68.8 kHz | 2310 |

Note that these filter constants apply only to the Input Modules – there may be other filtering in the signal path, not controllable via the User Interface, that would have to be taken into account to fully characterize the system.

Typically, a low filter value is used for a high frame rate, to maximize responsivity. A high filter value would go with a slower frame rate, to minimize noise.

**Examples:**

```
@FAM 3
```

Set all Input Modules to maximum filtering.

```
@FIM #1:0
```

Set Input Module "1" to minimum filtering.

```
@FAM?
@FAM! #0:1; #1:1
```

A query, and a response that indicates there are two input modules, both with a filter value of 1 (low).

```
@FIM? #2
@ERR^5
```

A query, and a response that indicates that Input Module "2" (the third module) is not present or detected.

# File Loading

Files that are resident in nonvolatile memory can be loaded into the sequencer RAM so that they can be run.

**FTC** Flash To Control

**FTP** Flash To Pattern

**Command syntax:**

```
@FTC
@FTP
```

Issuing either of these commands causes the corresponding Flash file to be copied to sequencer RAM. The previous contents of RAM, if any, are destroyed, but the Flash is not.

**Query syntax:**
There is no query or reply allowed for these commands.

**Reply format:**

```
@FTC!
@FTP!
```

These replies are returned automatically from the FTC and FTP commands, when the files are successfully copied. Unlike the complementary operations CTF and PTF, these copy operations are fairly quick, but still no command, query, or other communication should be attempted between the FTC/FTP command and the corresponding response. These replies are signals that the operation is complete.

**Errors:**
There are no errors returned for this command.

**Discussion:**
These copy operations are automatically performed at bootup, and are rarely needed. However if the user is experimenting with new sequences and merely wishes to restore the last ones saved with having to reboot, these commands will do the job.

**Examples:**

```
@FTC
@FTC!
```

The command and response show that the contents of the Control Flash memory were successfully copied to sequencer Control RAM.

## Bus Control

Two low-level commands are provided to read and write the system bus.

**ICR** Internal Control Read

**ICW** Internal Control Write

**Command syntax:**

`@ICR` *`addr; count`*

Read the device at address *addr*. The number of bytes to read is given by *count*.

`@ICW` *`addr; count; v0; … vn`*

Write to the device at address *addr*. The number of bytes to write is given by *count*. The data to write then follow.

**Query syntax:**
There is no query allowed for these commands.

**Reply format:**

`@ICR!` *`v0; … vn`*

In response to an ICR command, the number of bytes requested is returned.

**Errors:**
Error 5 is returned if the byte count is greater than the maximum allowed. In version 2.2.0, the maximums are 4 for ICR, and 6 for ICW. In later versions, the maximums are 12 bytes for either one. (Versions previous to 2.2.0 did not support these commands.)

**Discussion:**
These commands are for the benefit of users who build their own modules for the Little Joe camera. A discussion of the industry standard $I^2C$ bus is beyond the scope of this document.

**Examples:**

`@ICW $50; 2; $A0; 16`

A digital potentiometer at address 50 hex is commanded to write its wiper register (command A0 hex), the value 16.

`@ICR $9F; 4`

`@ICR! 238; 199; 30; 31`

A quad ADC at address 9F hex returns its four output values.

## Version Number

The version number of the Little Joe camera software is returned.

**JOE**

Little Joe

**Command syntax:**
There is no command syntax for this feature.

**Query syntax:**

```
@JOE?
```

This will query the version number.

**Reply format:**

@JOE! *a.b.c*

The version number is given in a.b.c format.

**Errors:**
There are no errors associated with this query.

**Discussion:**
The first digit ("a" above) is the major revision number. A change here generally indicates incompatibility with previous or subsequent versions. The second digit ("b" above) is usually incremented only when features or functionality are added. The third digit is incremented for minor fixes and tweaks.

As of version 2.2.0, this number is the only information returned with this query. Future revisions reserve the right to return other configuration information in formats yet to be determined.

**Examples:**

```
@JOE?
@JOE! 2.2.0
```

The camera is found to be running version 2.2.0.

## Data Conversion

This specialized command for certain types of Output Module controls the conversion of raw output data to an 8 bit or 12 bit format.

**MAP**

Map 10, 12, 14 or 16 bits onto 8 or 12 bits.

**Command syntax:**

`@MAP #m:v`

Right-shift the data in Output Module *m, v* bits.

**Query syntax:**

`@MAP? #m`

Query the mapping for Output Module *m*.

**Reply format:**

`@MAP! #m:v`

The map value of Output Module *m* is given in response to the query.

**Errors:**

MAP returns error 5 if *m* or *v* is out of range. Little Joe 2.2.0 does not detect the presence of Output Modules, as it does for Input Modules. Therefore a MAP command sent to a nonexistent module will not return an error as long as the module number is within the allowable range.

**Discussion:**

This command applies only to certain specialized Output Modules (OMs). The standard OM is set up to stream the whole data word, is not configurable, and will not respond. Also, this command is meaningless if a configurable OM is in the streaming mode (see the OMM command).

MAP controls the conversion of the raw output data to 12 bits for a demultiplexing OM, or to 8 bits for a display OM. The value *v* indicates how many bits the normally right-aligned data are shifted right into the smaller window. Zero (0) means that the LSBs are lined up.

For a 12 bit demultiplexing OM, shift values range from 0 – 4. For an 8 bit display OM, shift values range from 0 – 8, with additional values of 12 and 14 used to approximate gamma correction functions for 12- and 14-bit ADCs.

**Examples:**

`@MAP #0:4`

Set OM number 0 to shift right by four bits.

## Offset

There are two commands that pertain to setting the black level (offset). The offset of each channel can be controlled individually, or for convenience, all together.

**OAC** Offset All Channels

**OIC** Offset Individual Channel

**Command syntax:**

`@OAC v`

Set all channels to offset level *v*.

`@OAC >v`

Increase all channel offset values by *v*.

`@OAC <v`

Decrease all channel offset values by *v*.

`@OIC #c:v`

Set offset of channel *c* to value *v*.

`@OIC #c:>v`

Increase offset of channel *c* by *v*.

`@OIC #c:<v`

Decrease offset of channel *c* by *v*.

**Query syntax:**

`@OAC?`

Query the offset values of all channels.

`@OIC? #c`

Query the offset value of channel *c*.

**Reply format:**

`@OAC! #0:$v; #1:$v; … #n:$v`

List the offset values of channels 0 – *n* (as many as are present in the system, up to 16). Each channel is listed individually, even if all values are the same.

`@OIC! #c:$v`

The offset value *v* of channel *c* is given in response to the query.

**Errors:**
OAC will return error 5 if *v* is out of range. OIC will return error 5 if *v* is out of range, or if *c* is greater than or equal to the number of channels present. However if usage of the increase/decrease syntax (“>” or “<”) takes a value out of range, the offset will be adjusted as far as possible in the indicated direction, and no error is returned.

**Discussion:**
The offset can have a value of 0 – 1023, inclusive. The offset determines the black level of the image; the greater offset makes the darker image. The offset value is a relative setting, in that the actual number has no meaning to the user. Rather, the separate channels can be balanced with each other to compensate for variations elsewhere in the system, and the overall brightness adjusted for the best image.

The offset value (like any parameter) can be specified in decimal, hexadecimal or binary format, with the proper prefix. But notice that the reply format uses a mixed format: decimal for the channel numbers, and hexadecimal notation for the values, with the “$” prefix and six digits, including leading zeros. The Little Joe tends to represent numbers less than 256 in decimal form, and prefers fixed-field hexadecimal for larger numbers.

**Examples:**

`@OAC 512`

Set all channel offsets to a midrange level.

`@OAC $200`

Set all channel offsets to a midrange level (200 hex = 512 decimal).

`@OIC #0:>10`

Increase the offset of channel “0” by 10.

`@OAC?`

`@OAC! #0:$0003FF; #1:$0002FF; #2:$0001FF; #3:$000123`

A query and response that indicate there are four channels. Channel “0” has a value of 1023 (3FF hex), “1” has a value of 767 (2FF hex), “2” has a value of 511 (1FF hex), and “3” has a value of 291 (123 hex).

## Output Module Mode

Controls the operation of certain specialized Output Modules.

**OMM**

Output Module Mode

**Command syntax:**

`@OMM #m:v`

Set Output Module *m* to mode *v*.

**Query syntax:**

`@OMM? #m`

Queries the operating mode of Output Module *m*.

**Reply format:**

`@OMM! #m:v`

The operating mode code of Output Module *m* is given in response to the query.

**Errors:**

OMM returns error 5 if *m* or *v* is out of range. Little Joe 2.2.0 does not detect the presence of Output Modules, as it does for Input Modules. Therefore an OMM command sent to a nonexistent module will not return an error as long as the module number is within the allowable range.

**Discussion:**

For configurable Output Modules, a value of zero (0) means streaming mode, and a value of one (1) means demultiplexing.

**Examples:**

`@OMM #4:0`

Puts Output Module 4 into streaming mode.

`@OMM? #0`

`@OMM! #0:1`

A query, and a response that indicates Output Module 0 is in demultiplexing mode.

# Program Selection

Little Joe stores up to eight programs. A program is a set of low-level command to the CCD, that typically determines frame rate, binning, image size, etc.

**PRG**

Program

**Command syntax:**

`@PRG n`

Select program 0 – 7.

**Query syntax:**

`@PRG?`

Query, which program is currently loaded.

**Reply format:**

`@PRG! n`

The index of the currently loaded program is given in response to the query.

**Errors:**

PRG will return error 5 if *n* is greater than 7.

**Discussion:**

Because the Little Joe is programmable, it is not definitive to state what each of the eight programs will do. Not all programs may be present. That understood, it can be stated that the Standard Program Set 2.0 has these characteristics:

| Program | Nominal Frame Rate (Hz) | Image size | Binning |
|---|---|---|---|
| 0 | 1000 | 80 x 80 | 1 x 1 |
| 1 | 625 | 80 x 80 | 1 x 1 |
| 2 | 125 | 80 x 80 | 1 x 1 |
| 3 | 40 | 80 x 80 | 1 x 1 |
| 4 | 2000 | 40 x 40 | 2 x 2 |
| 5 | 3000 | 40 x 40 | 2 x 2 |
| 6 | 5000 | 26 x 26 | 3 x 3 |
| 7 | 2000 | 80 x 80 | 2 x 2 |

Note that in this set, the actual frame rates are slightly (approximately 0.1%) faster than nominal, to allow for hardware synchronization.

PRG does not affect any operating parameters. If the program number is changed while the camera is running, the old program will stop and the new one will commence at its beginning. See the related commands SEQ, TXC, SAV and RCL.

**Examples:**

```
@PRG 0
```

Select program 0. If the camera is running, it will continue to run with the new program. If stopped, it will stay stopped until started.

```
@PRG?
@PRG! 5
```

A query, and response that indicates program 5 is loaded.

```
@PRG 8
@ERR ^5
```

An attempt to load an invalid program number results in error 5.

## Quiet Mode

This command allows the boot-up messages to be turned on or off. Implemented in version 2.2.2.

**QUI**

Quiet mode.

**Command syntax:**

`@QUI` *v*

*v* is 1 to turn Quiet Mode on, or 0 to turn it off (default).

**Query syntax:**

`@QUI?`

Query the mode.

**Reply format:**

`@QUI!` *v*

The value *v* will be 0 if Quiet mode is off, or 1 if it is on.

**Errors:**

QUI returns error 5 if a value other than 0 or 1 is passed as a parameter.

**Discussion:**

Normally at boot-up, several status messages are displayed. If these messages are not desired, they can be silenced by putting the camera into Quiet mode. As with the baud rate, this command will only be meaningful if the SAV 0 command is issued after setting the mode. Typically, the user would issue RCL 0, QUI *v*, SAV 0 to set Quiet mode without disturbing the other parameters in set 0.

**Examples:**

```
@QUI?
@QUI! 1
```

A query, and a response that indicates that the camera is in Quiet mode.

```
@QUI 1
```

Set the camera to Quiet mode.

## Save/Recall Settings

Most of the camera's operating parameters can be stored in one of eight nonvolatile registers to be recalled easily later. These commands are complementary to each other.

**RCL** Recall settings.

**SAV** Save settings.

**Command syntax:**

`@RCL` *v*

`@SAV` *v*

The value *v* is a number from 0 – 7, for one of the eight parameter storage registers.

**Query syntax:**
There is no query available for these commands.

**Reply format:**

`@RCL!` *v*

RCL returns with this message to signal that it is complete.

**Errors:**
Error 5 is returned if *v* is out of range, or if a query is attempted.

**Discussion:**
As of version 2.2.0, SAV saves (and RCL recalls) these parameters:

- Attenuation
- Filtering
- Offsets
- Signal delays (clamp & sample)
- Accumulator repetitions
- Program selection
- Data conversion
- Output module mode
- Run/stop/remote mode
- Baud rate (SAV 0 only)

The contents of register 0 are automatically loaded at bootup.

Typically, there is one-to-one mapping of parameter registers and programs, but this is by no means necessary. Any program number (or indeed any legal parameter) can be stored in any register. This flexibility can also lead to confusion.

In a typical Graphical User Interface, all of these parameters would have to be queried after a RCL command, to ascertain what they are. There is no "pushing" of information with these commands.

Note that SAV does *not* save the control or pattern sequence files. Those operations are part of file management, and are handled by CTF and PTF (q.v.).

**Examples:**

```
@SAV 0
```

Save all the above-mentioned parameters in storage register 0.

```
@RCL 7
@RCL! 7
```

The contents of register 7 are recalled and confirmed.

## Repetition

Adjust the number of times the accumulation segment of a program is repeated.

**REP** Repetitions

**Command syntax:**

`@REP n`

*n* is the number of *additional* accumulator repetitions to add to the one that is always present by default. It can be any positive number from 0 – 65535. The total number of iterations is *n* + 1.

**Query syntax:**

`@REP?`

Query the number of repetitions in the accumulator register.

**Reply format:**

`@REP! $00hhhh`

REP always replies in six digits of hexadecimal notation. Since the accumulator is a 16 bit register, the first two digits are always 00. This is the reload value of the register; thus it will not return the number of repetitions already elapsed, or yet to play.

**Errors:**

REP ought to have error checking, but as of version 2.2.0, it does not. Values greater than 65535 will "wrap around" and yield modulo 65536. Users should not depend upon this behavior, as it is expected that some future version will have range checking, to return the standard error 5.

**Discussion:**

The base frame rate can be slowed, and lower light levels accommodated, by adjusting the accumulator. Every program loaded in the Little Joe is preceded by an accumulator (or integrator) segment, which allows the CCD to collect light until it is time to read the pixels. This segment runs at least once, and may be run up to 65,536 times.

**Examples:**

`@REP?`

`@REP! $000004`

A query, and a response that indicates that there are five total iterations of the accumulation segment of the current program.

## Sequence

This command starts or stops the camera for image acquisition.

**SEQ** Sequence control

**Command syntax:**

`@SEQ` *v*

The value *v* is 0 (zero) to stop the sequence, or any other value to start it.

**Query syntax:**

`@SEQ?`

Discover whether the camera is running.

**Reply format:**

`@SEQ!` *v*

The value *v* will be 1 if the camera is running, or 0 if stopped.

**Errors:**

There are no errors returned for this command.

**Discussion:**

This command starts or stops the camera for image acquisition. It stops the sequence immediately (within the limits of the software latency), so the state of the sequencer will be indeterminate when the SEQ 0 command is issued. Starting, however, causes a reset and causes the current sequence to start at the beginning. Typically, this means the start of the first pixel acquisition.

If the camera is under control of the Remote Synchronizing Interface, the response to the "SEQ?" query will be meaningless, since the camera is no longer under software control.

**Examples:**

`@SEQ 1`

Start the camera.

`@SEQ?`

`@SEQ! 0`

A query, and a response that indicates the camera is stopped.

## Temperature

Primarily, this command is used to query certain operating temperatures within the system. In systems where dewar temperature can be set or controlled, this command can also be used to control the set-point.

**TMP** Temperature

**Command syntax:**

`@TMP v`

Set the dewar (or heater) temperature controlling DAC to value *v*.

**Query syntax:**

`@TMP?`

Query the values returned by the system's ADCs.

**Reply format:**

`@TMP! #0:v0; #1:v1; #2:v2; #3:v3`

The values *v0* through *v3* are the values reported by the four internal ADCs.

**Errors:**

There are no error messages particularly associated with TMP.

**Discussion:**

The temperatures returned by the query are the raw numbers returned by the Analog/Digital Converters (ADCs) that serve the temperature sensors in the camera. The first value (*v0* above) represents the case temperature. Its sensor is on the Service Module, and the temperature in C is determined by the formula:

$$(1/(a + b(\ln(v0/207)))) - 273$$

where $a = 3.354 \times 10^{-3}$

$b = 2.888 \times 10^{-4}$

*v0* is the value returned by TMP.

Assuming a CCD39 (Peltier) build, values v1 and v2 represent the two sensors on the CCD itself. Here the temperature in C is given by:

$$(c/((\ln v/2.55) + d)) - 273.15$$

where $c = 3725.6$

$d = 11.403$

v is the v1 or v2 value returned by TMP.

Value *v3* is designed to be used with a vacuum sensor, if present. The conversion formula is undetermined at this time, and the value should be ignored.

Similarly, until a system utilizing a heater is built, a conversion formula for the TMP set-point command is undetermined.

**Examples:**

```
@TMP?
@TMP! #0:211; #1:63; #2:51; #3:238
```

The temperatures were queried, and the four values returned were: case 24.7° C, CCD -18.1° and -14.4° C; the fourth value is ignored.

## External Control

This command controls the operation of the Remote Synchronizing Interface (RSI).

**TXC** Take External Control

**Command syntax:**

`@TXC` *v*

The parameter *v* may be 0, 1, or 2. Zero (0) transfers the camera to local control, so the RSI has no effect. A value of 1 uses the RSI in "freeze" mode, and 2 uses it in "repeat" mode (see discussion below).

**Query syntax:**

`@TXC?`

Query the status of the RSI.

**Reply format:**

`@TXC!` *v*

The value returned in *v* is 0, 1, or 2, as discussed below.

**Errors:**

TXC returns error 5 if a value other than 0, 1, or 2 is passed as a parameter.

**Discussion:**

When under local control, the RSI has no effect on camera operation. When the camera is put into external control mode, the pins of the RSI have these functions:

| Pin | Name | Function |
|---|---|---|
| 1 | Gnd | Ground; signal reference. |
| 2 | Stop | Bring to ground to run; float or drive high (TTL or +5v) to stop. |
| 3 | Pr0 | Pr2, Pr1, and Pr0 are binary code inputs for the program number. |
| 4 | SOS | Start of Sequence; an output pulse. |
| 5 | Pr1 | (See Pr0) |
| 6 | SOA | Start of Accumulation; an output pulse. |
| 7 | Pr2 | (See Pr0) |

The RSI is discussed in detail in the Harware Interface section of this document. In brief, the TXC command determines how the sequence will terminate in response to the Stop signal being asserted. In "freeze" mode, the sequence runs its course and stops at the end, staying there until Stop is released. In "repeat" mode, the sequence runs its course as before, but repeats the accumulation segment indefinitely until Stop is released.

TXC also controls how programs are selected. For TXC 0, programs are selected exclusively by the PRG command. For TXC 1 or 2, the RSI pins Pr2, Pr1 and Pr0 aid in the selection. When all three pins are driven (or allowed to float) high, then the program is selected by the PRG command as before. However if any of these pins are brought low, then the program is selected by the binary code represented by the three bits.

Thus if all are low, program 0 is selected. If Pr0 is high while the others are low, program 1 is selected, and so on. If all are high, program 7 is NOT necessarily selected, as might be expected. Instead, program selection is transferred to the PRG command.

The "PRG?" query will work properly under remote control, and the "SEQ" command can be used to start and stop the camera, but the "SEQ?" query will not work, because the software can not monitor the status of the Stop line of the RSI.

**Examples:**

```
@TXC 1
```

Put camera under control of RSI, in "freeze" mode.

```
@TXC?
@TXC! 0
```

Camera responds to a query that it is under local control.

```
@TXC 3
@ERR ^5
```

Error 5 is returned to an invalid parameter value.

## File Transfer

Sequence files are uploaded to the camera via the common Xmodem protocol.

**XMC** Xmodem Control File.

**XMP** Xmodem Pattern File.

#### Command syntax:

`@XMP`

`@XMC`

These commands start the file transfer to the appropriate part of the sequencer memory. They do not respond with ACK like most other Little Joe commands; instead they go immediately into the Xmodem/CRC protocol, echoing 'C' characters at intervals.

#### Query syntax:

There is no query or reply allowed for these commands.

#### Reply format:

`@XMO! $`*`hhhhhh`*

This is not a reply as such. Rather it is a signal that the file transfer is complete. The number returned is the number of packets (128 bytes) transferred. It will be a hexadecimal number of six digits with leading zeroes.

#### Errors:

These commands will return errors 101 through 110 upon failure of the transfer. The exact error number is meaningful only to SciMeasure technical support, as they represent failures at different stages of assembly language code.

#### Discussion:

There are two types of files used by the Little Joe sequencer. Pattern files are 24 bits wide, and Control files are 8 bits wide. A sequence requires (and consists of) one of each type. Files sent to the CM are put directly into the Pattern RAM or Control RAM, at which time they are ready to run. Files can not be written directly to Flash; they must go to the appropriate RAM first and then be copied (see the CTF and PTF commands). See the Technical Manual for more information on these memory structures.

File transfer is via XModem or XModem/CRC (preferred). Since the AIA protocol does not support this, it is only guaranteed to be available through the Command Module port. If a third-party system is driving the camera through the Output Module, XModem transfer might not be available. Variants such as YModem and XModem-1K are not supported.

The file name is not transferred along with the data. A full discussion of Xmodem/CRC is rather beyond the scope of this document.

Chapter 3

# A Graphical User Interface

*Presenting a Windows-based GUI that can be used to control the Little Joe camera.*

The command interface to Little Joe is based on ASCII text commands, and can be accomplished with a standard terminal or emulator. However, this can become tedious if a number of adjustments are to be made. To make controlling the camera easier, a graphical front end can be spliced onto a terminal emulator engine. This allows control with familiar mouse clicks, and no memorization of command syntax. An example screen shot of such an interface is shown here.

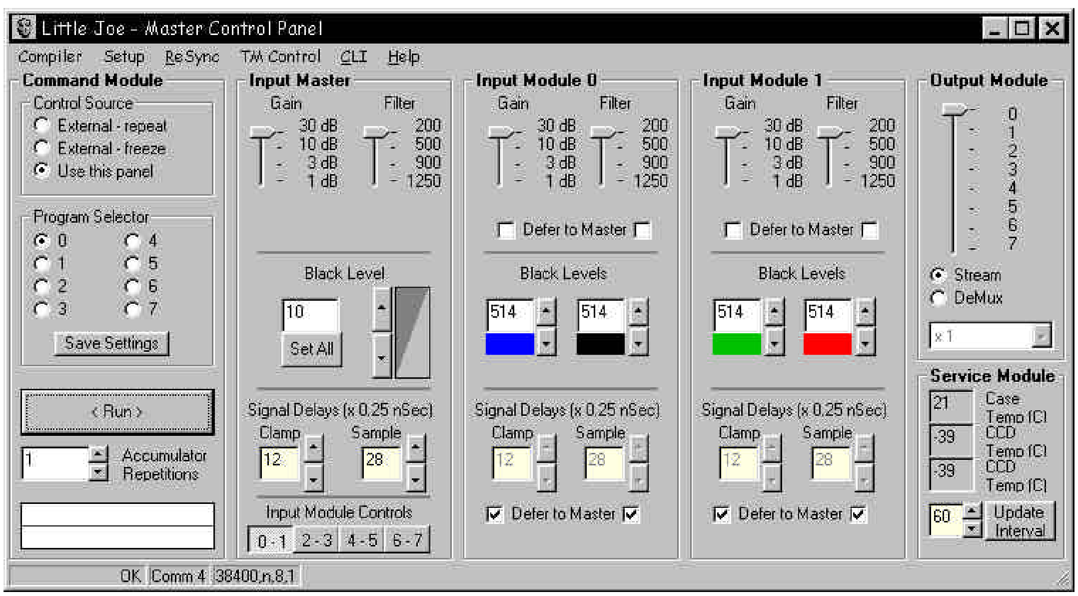



**FIGURE 7.** *A Master Control Panel for a universal controller interface. This provides total access to the camera, including features not generally used.*

This interface is probably not the best for an end user, because it allows so much control that it becomes cluttered and potentially confusing. However, it is versatile, and more importantly, it is available.

## *Installation*

Probably the most difficult part of using this program is getting it started the first time. It ships as a self-installing setup executable, and resides in "C:\Program Files\Little Joe" or on another drive with the same path, if "C:\" is not the default. The installation is shipped as a "Setup" program and CAB files.

When the GUI is launched, it looks for the file "LittleJoeGUI.ini" in the same subdirectory (or folder) where the GUI resides. This file tells the program how to communicate with the camera. If the file is not found, the program assumes a new installation and tries to detect the camera, first on the AIA port (Output Module interface), then, failing that, on the computer's serial ports (Command Module interface). In the latter case, it presents a setup screen where the serial port and baud rate can be selected. The baud rate should be set to 38400 unless you know that it is less, in your particular installation. If the file "LittleJoeGUI.ini" does not exist, the program will create it, and save the operating parameters in it.

**Note**: when using the AIA port, it is usually necessary to install hardware-specific device drivers to support the AIA interface card. Sometimes the drivers supplied with the cared are newer than those shipped with this GUI, rendering the GUI incompatible. The solution is to copy the newer versions into the folder where the GUI resides. For more specific instructions, see Chapter 4 (Applying the Camera).

If the camera is powered on, the control panel shown in Figure 7 should appear. If it does not, there will be a small "splash" screen instructing the user to power the camera, wait 15 seconds, then hit the "Ping" button. If the control panel still does not appear, there is likely an error in the hardware connection. See Chapter 1 (The Hardware Interface) of this Users' Guide for possible remedies.

It may also be necessary to manually edit the LittleJoeGUI.ini file to overcome some of the application's default assumptions. Under the [Comms] heading there are three keys: Door, Port, and Baud. Here is an example excerpt:

```
[Comms]
Door=Front
Port=4
Baud=6; 0=6,1=12,2=24,3=48,4=96,5=192,6=384
```

Door=Front means to use the CM port, while Door=Back means the OM port. The Port key may not be present if Door=Back, otherwise it specifies the Comm port used for the CM interface. Port=1 means Comm 1, etc. Baud=6 indicates 38400. Other possibilities are shown in the comment line.

When the control panel opens, the program then queries the camera for all of its settings, and displays the responses as control settings. In this way, the panel reflects the true state of the camera.

## *Operation*

The following discussion relies heavily on the terminology and mnemonics put forth in Chapter 2, "The Command Set." We will discuss each section of the control panel, frame by frame, then the menu bar. Almost all controls have a "tool tip" which will appear next to the cursor, giving extra hints or reminders as to the control's usage or max/min values.

#### Command Module

The Control Source frame issues the TXC command: TXC 2 for repeat, TXC 1 for freeze, and TXC 0 for local control. The Program Selector issues the RCL command when the button is clicked. It also queries all the settings, much the same as the "ReSync" selection on the menu bar (see below). Like ReSync, it can take some time to complete, depending on the baud rate. Save Settings issues the SAV command for the button that is already selected. The <Run> button issues the SEQ command. For the Accumulator Repetitions, the user can use the spin buttons for fine control, or just type in the desired number. The blank boxes at the bottom of this frame display the results of the CTF and PTF queries.

#### Input Master

Gain and Filter issue the AAM and FAM commands, respectively. When either of these is manipulated, the individual Gain and Filter controls will automatically track them. The labels on these controls (30 dB, etc.) are only approximate. Black Level is used to issue OAC. Type a number *x* into the box and hit Enter or Set All to send OAC *x*. Use the spin buttons to send OAC >*x* or OAC <*x*. Again, the individual module controls will track. The Signal Delay controls send DCA and DSA, but here only the spin buttons work. Lastly, the "Input Module Controls" set determines which two Input Modules will have their controls displayed on the panel. This does not invalidate or disable the other modules, it just helps keep the control panel from becoming overwhelming.

#### Input Module x

Gain and Filter send the AIM and FIM commands, respectively. They will work only if the Defer to Master box is unchecked. The Black Level controls send OIC *x* if a number is entered (and you *must* hit Enter), or OIC >1 and OIC <1 if the spin buttons are used. The Signal Delay controls send DCI and DSI, but again, only if the Defer box is unchecked.

#### Output Module

The slider control determines which Output Module will receive the next command; by itself it doesn't issue anything. Stream sends OMM *m*: 0, and DeMux sends OMM *m*: 1. If the module is in DeMux mode, the selector box below issues the MAP command. The selection "÷4" issues MAP *m*: 0, "÷2" issues MAP *m*: 1, and so on down the list. Admittedly, this is not very intuitive; the grammar is a holdover from an earlier product. *Please note that most Output Modules will not receive or respond to commands. This control box is useful only on modules built as "Multi-mode."*

#### Service Module

The three temperature boxes display the results of the TMP query. Two separate CCD temperatures indicate a CCD equipped with two sensors. The TMP query is issued periodically, according to the number of seconds set in the Update Interval box. Sometimes it is desirable temporarily to suspend these queries; clicking on the Update Interval button will do it.

**Menu Bar**

- Compiler
    - *Edit* simply launches the default text editor, such as Notepad.
    - *Compile* is not implemented. It just launches a DOS box.
    - *Upload Control* sends the XMC command after launching a dialog box that lets the user pick which binary control file to upload.
    - *Upload Pattern* is the same, except with the XMP command.
    - *Keep Control* sends CTF after prompting for a name.
    - *Keep Pattern* similarly sends PTF.
- Setup launches the setup dialog box, allowing the user to set the Comm port and baud rate. It may be necessary to close and re-open the program after changing these settings. The "Monitor Transactions" checkbox opens a small window on the bottom of the control panel to allow the user to see what is happening on the communications link, for troubleshooting purposes.
- ReSync queries the camera for all its operating settings. If the camera suffers a power failure, or if the communications link is interrupted, the camera settings can change, without the control panel reflecting the changes. This item will *not* change the camera settings, it only causes the GUI to show what they really are. It does this by issuing all necessary query commands, one at a time, and waiting for each response in turn. Note that this behavior causes it to take a few seconds to complete, as each command is processed. Lower baud rates can have a large effect here.
- TM Control launches a dialog box that allows control of the Timing Module.
- CLI launches a Command Line Interface dialog box. The user can type in any command. This is useful if a new command is implemented in the camera but not present on the GUI, and for troubleshooting. This interface does not automatically prepend the "@" symbol to messages; it just sends them straight out.
- Help
    - *Contents* launches the Help file program. At present, this is mainly just tips on getting connected, and doesn't have anything not in this Users' Guide.

- *About* will display the version numbers of the GUI, the CM, and other components, if known.

**Disclaimer**

The GUI was developed by SciMeasure for our own use, on our own development systems. Due to the wide variety of computers and OS's in the world, it is not guaranteed to work universally. It has been known to work on Windows 95, 98, NT and 2000. Your results may vary.

Chapter

4

# Applying the Camera

*Typical installation and usage are discussed, giving concrete examples and instructions.*

One problem with a product that is as versatile and purposely platform-independent as Little Joe, is that difficult to enumerate all the possible configurations. To avoid locking in to a few "standard" setups, we tend to describe the interface in broad terms and generalities, leaving the user to think (perhaps) "fine, but how do I set up *my* system?" There are also the complementary dangers of either missing something crucial, or getting bogged down by imagining the system to be more complicated than it really is. This chapter should help the user by reviewing all the steps need for a typical installation, perhaps showing the way to whatever adaptations may be needed for a particular use.

*Assemble the camera.*

The Little Joe consists of a camera head, a controller, and a power supply. There is a bundle of cables that connects the head to the controller, and a separate data cable to connect the controller to the host computer. Not part of the Little Joe, but still essential to its operation, is some kind of device to receive, process and interpret the data. Typically, this "device" is a Frame Grabber card (FG) that plugs into a PCI expansion slot within the host computer. Also, most installations require a computer to issue commands to the Little Joe, in order to control it. This is usually the same computer that receives the image data.

The first step is to mount the camera head, securely, where it will go. It is rather cumbersome to manipulate once the cables are attached, and is fairly delicate. In operation the head radiates about five watts, so it doesn't need a lot air flow, but neither should it be completely contained.

The position of the head, and the length of the cables, limits where the controller may go, so the next step is to place that. A desktop unit has its own ventilation fans, but it needs a free flow of air all around it for them to work. The air intake is under the unit, and the exhaust is at the back, so these areas must be kept free of obstruction. A rack-mount unit has no fans, and depends on free air circulation within the rack. A heavily loaded rack should provide forced air through the Little Joe controller. The power supply, if separate, may also be placed and connected at this time.

**Make certain that the power is OFF before proceeding to hook up the head.** Catastrophic failure is almost certain, if the head is only partially connected while powered.

Now the head may be attached to the controller, using the cable bundle. The bundle is constructed with different connectors on each end, so there is only one way to connect it. It is usually easiest to start with the BNC connectors at the controller end. They are color coded to the Input Modules. The Service Module and Clock Driver Module cables should be plugged in and secured using the slide-locks supplied. Take the extra time to ensure that the lock has clicked firmly into place at both top and bottom of the connector.

FIGURE 8. *The Service Module cable is attached properly but not yet locked down. Note the loop at the top of the connector.*

FIGURE 9. *The Service Module cable is now properly locked in place. Note the new positions of the loops of the slide lock.*

FIGURE 10. *A close-up shows that the bottom of the connector is secure.*

FIGURE 11. *The slide lock has clicked into place, but is not secure on both posts. Do not permit this to happen.*

Similarly, make the connections at the camera head. Again, note the color code of the BNCs, and double check the slide locks.

The last step would be to connect the image data cable or cables to the processing computer. Normally this is the SCSI-3 cable that connects the Output Module to the Frame Grabber. Use whatever strain relief mechanism is appropriate.

**Variations**

The one-computer system described above is probably the simplest version. In one common variation, the device that receives the data is not a frame grabber, but a Digital Signal Processor (DSP) for controlling an Adaptive Optics system. Such a system may have two Output Modules (one for an FG, one for a DSP); or it may forgo the FG connection, in which case it would still need at least a terminal to send commands to the Little Joe.

Another common variation puts the controller into a rack-mounted case with an integral power supply. It is possible to fit two Little Joe controllers into a single rack.

The camera head, too, may be supplied in a variety of form factors, depending on the mounting requirements of the user.

*Install a frame grabber.*

The frame grabber used most often with the Little Joe is the PCI-DVK, made by EDT, Inc. It is not a SciMeasure product, so any technical questions should be taken to the manufacturer (http:\\www.edt.com).

The PCI-DVK installs on a local PCI bus, and is usually supplied with an installation manual as well as a software package. Little Joe does not require any variation in the manufacturer's procedure.

*Install the software.*

The software has two components: image processing, and camera control. These are probably the most variable aspects of the system, so they can't be treated in much depth here. However, they are also most likely the parts that the user is most familiar with, so the lack of depth should not be a problem. Either or both of these components may be supplied by the user, or by a third party who furnishes the documentation. The PCI-DVK requires a number of device drivers and related components, so these should be installed according to directions.

**Note:** Sometimes these drivers are updated in such a way as to render the control software incompatible with the hardware. Particularly, the "Little Joe GUI" program uses its own copy of pdvlib.dll to communicate via the AIA port. If communications are a problem, it may be necessary to copy all the *.dll files from "C:\edt\pdv\" into "C:\Program Files\Little Joe\".

For image processing software, a good demonstration program is provided with the EDT frame grabber, and it is installed along with the drivers. It is called "PdvShow" and is installed in the "C:\edt\pdv\" subdirectory. However, the installation lacks certain configuration files necessary to operate the Little Joe. These files are supplied by SciMeasure, and must be installed in "C:\edt\pdv\camera_config\". These files are called

Lil-Joe-40x40.cfg, Lil-Joe-64x64.cfg, and Lil-Joe-80x80.cfg. There may be others, depending on your specific configuration, and they would be supplied when the camera is delivered.

The other software component, camera control, may be integrated with the display package from a third party, or may be something like the GUI discussed in Chapter 3 of this Users' Guide. Or it may be, as stated elsewhere, a simple terminal emulator. In any case, the user interface has been discussed in other chapters.

*Make it go.*

With everything installed, it only remains to power the camera and launch the software. It can take up to 15 seconds for the camera to boot, so don't expect it to respond to commands immediately.

When using PdvShow, it is best to launch that before attempting to start the communications program. This is because PdvShow initializes the PCI-DVK board, a necessary step before a program like the Little Joe GUI can launch. In PdvShow, go to the menu bar and select Camera/Setup. This opens a dialog box where you can select the SciMeasure camera in whichever configuration suits the program you wish to run. Assume you are using the SciMeasure standard program set 2.0. (See Chapter 2, "The Command Set" under the heading Program Selection PRG.) Program 0, the default, has an image size of 80 x 80, so select that configuration in PdvShow.

Now you can launch the GUI, and the camera should respond. In PdvShow, select Camera/Continuous Capture, and the image should appear. PdvShow provides many features that can manipulate the image; consult the PdvShow documentation.

#### Disclaimer

The procedures outlined in this chapter have been shown to work on many systems. However, computers and applications being what they are, there may be additional steps required before the system is fully functional. However, the Little Joe is not particularly demanding, and as long as the host computer can handle the data flow put out by the frame grabber, there should be no insurmountable problems.