Satshell Client - User Guide

OMNIFLAGS INC. CLASSIFIED DOCUMENTS

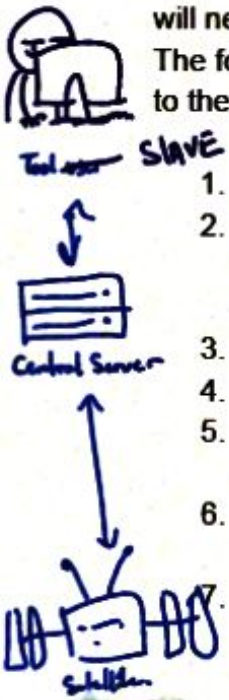
Introductory Letter

Hello Valued Employee! ♥ ♥ ♥ con, your corporate overlords ♥ ♥ ♥

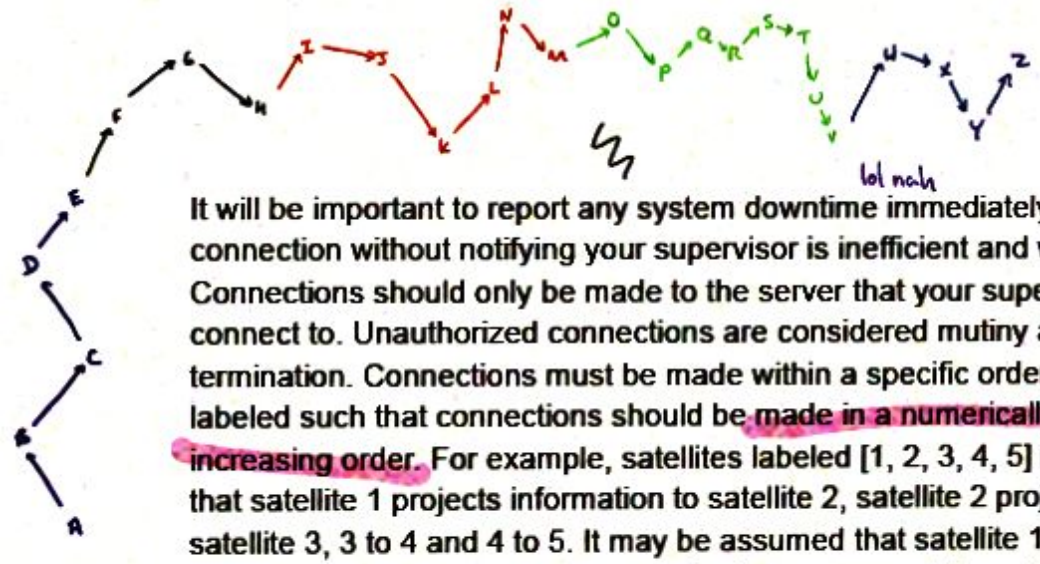
Receiving this user manual solidifies your position as an OMNIFLAGS INC. Communication Servicer. Your core responsibility will be ensuring communication between the central communication server and the extra-terrestrial dataserver. This user manual should provide necessary information for you to manage this connection and ensure the efficient flag distribution our company provides. Your work is integral to the future of our company, and we expect great results from your job term. Any questions should be asked to your employer directly via verbal communication.

Enforcing the connection

As a Communication Servicer, your responsibility is to establish a connection between the central communication server and the extra-terrestrial dataserver. Your supervisor should have provided the appropriate IP address and port you will need to configure your client program as necessary.
The following steps will describe the process to update your client's connection to the central communication server.



1. Log-in to your OMNIFLAGS employee account
2. If you are initializing your workspace for the first time, click on "Configure". Otherwise, navigate to "Options > Network > Connection to Central Server > Reconfigure"
3. Within the "Host" input field, enter the IP address as provided by your employer
4. Within the "Port" input field, enter the port value as provided by your employer
5. Ensure your connection is properly established, start the client's built-in debug tool. This will open the debug terminal along with the list of commands to use.
6. Run a test command by entering "POSN" into the debug terminal. This should return with an appropriate output.
7. Confirm your changes by clicking "Confirm"

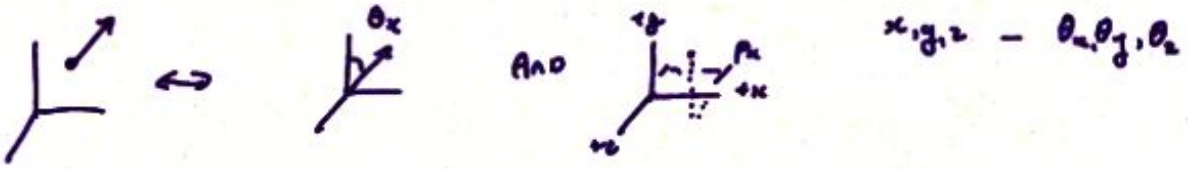


lol nah

It will be important to report any system downtime immediately. Failure to establish a connection without notifying your supervisor is inefficient and will result in termination. Connections should only be made to the server that your supervisor instructs you to connect to. Unauthorized connections are considered mutiny and will result in termination. Connections must be made within a specific order. Satellites are typically labeled such that connections should be made in a numerically or lexicographically increasing order. For example, satellites labeled [1, 2, 3, 4, 5] must be connected such that satellite 1 projects information to satellite 2, satellite 2 projects information to satellite 3, 3 to 4 and 4 to 5. It may be assumed that satellite 1 and satellite 5 already communicate to the central communication server and the extra-terrestrial dataserver respectively. Establishing this communication enables a bidirectional datastream between the two servers. This allows all OMNIFLAGS employees to browse all shared files. [illegible] do not write on company documents -Management

When pinging the positions of satellites, an arbitrary set of axes are defined such that the y-axis points from the central communication server to the extra-terrestrial dataserver. The scale of these axes are made arbitrary. Using these axes, the satellites to connect will be displayed in 3D space within the Graphical Satellite Display tab. Satellites may be moved or reoriented to establish connections as necessary. The Graphical Satellite Display provides necessary information to assist with this process. [illegible] within the grid represents a satellite, each is labeled with the symbol

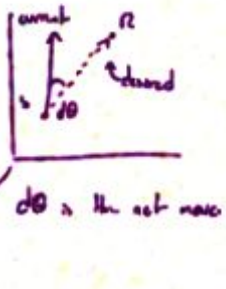
[illegible] computational energy and [illegible] properly



Outward-pointing arrows show the orientation of the satellite, and should be turned towards the next satellite to connect to as necessary. Changes in position and orientation will be done with respect to each set of axes. That is, calculations of positions or orientations are done along with the arbitrarily defined set of axes. That is, for a given orientation vector, the X-component of the satellite's angular orientation is the angle between the vector and the X-axis. Likewise, the X-component of the satellite's position is the distance of the satellite from the YZ-plane along the X-axis.

The Satshell Client tool will ensure that calculations of any differences will be determined correctly with respect to the Cartesian coordinate systems. Simply adjust each knob within within the Control Panel tab, or enter specific values within the input fields below each knob, to reorient the satellite as necessary. Satellite orientations must point to the next satellite within one (1) degree of uncertainty. Exceeding orientations of one (1) degree risk faulty signals and the failure to establish a connection. The Graphical Display tab will emulate changes as necessary in real-time to aid in making accurate changes. These changes do not affect the Satellites in real-time. To perform these changes, the "Execute" button must be clicked. This will inform the Satshell Client to compile the necessary machine code to the central communication [illegible] communicates with each satellite as necessar[illegible]

5. Ensure position of satellites are as desired
6. Click "Execute" to compile and send necessary machine code



As mentioned, satellites may also be reoriented. The following instructions explain this process.

Reorientation of Satellites

1. Refresh orientations to ensure validity of adjustments
2. Adjust "Angle-X" dial until the orientation vector is sufficiently angled from the X-axis
3. Adjust "Angle-Y" dial until the orientation vector is sufficiently angled from the Y-axis
4. Adjust "Angle-Z" dial until the orientation vector is sufficiently angled from the Z-axis
5. Ensure orientation of satellites are as desired
6. Click "Execute" to compile and send necessary machine code

Note: Satellite orientation must be within one (1) degree of the expected vector to ensure successful communication.

why your libraries don't use this ???

Custom Architecture

As a core part of debugging, it is important to note that the satellites runs off a custom, proprietary architecture. The following block diagrams of the processor are simplified to contain the components necessary to complete each task as necessary.

Full Architecture Reference

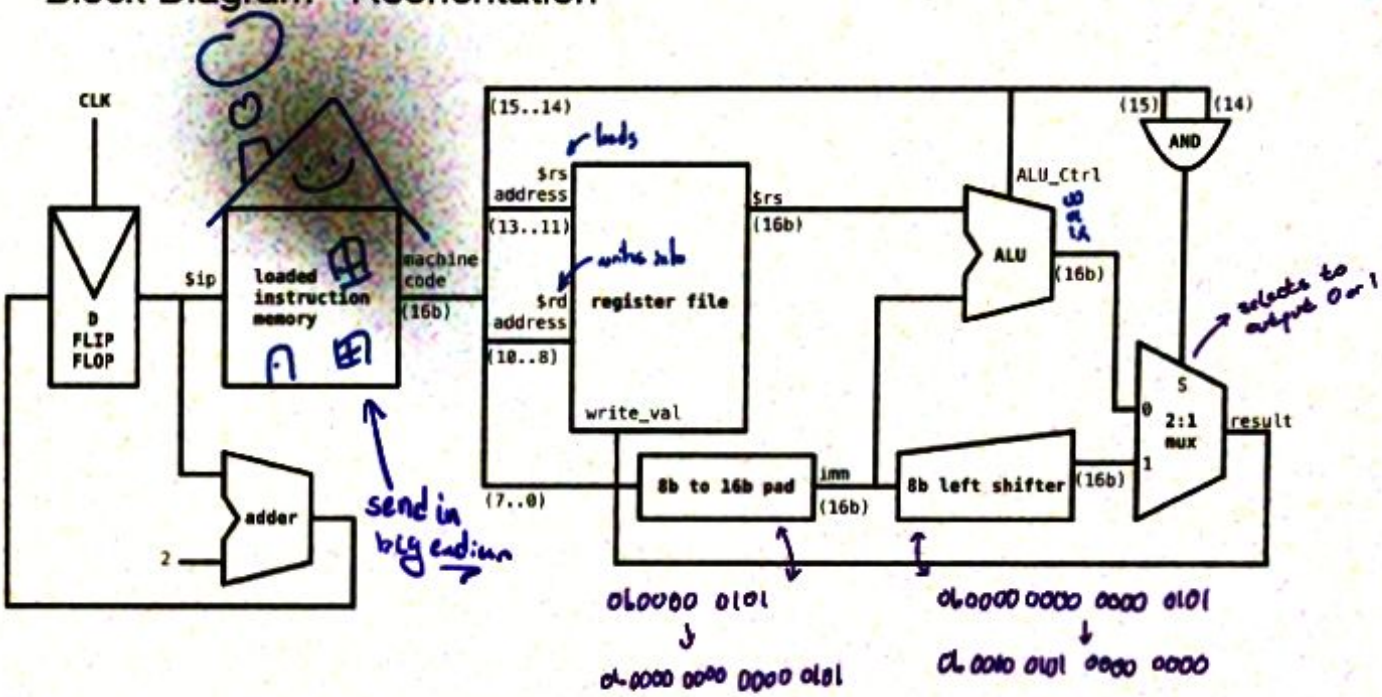
Reference Document	Page Number
Table of Contents	11
Architecture Registers	12
abs	12
adc	12
add	12
addi	12
	13

> speak memory to make your own processor
> crash a library which will only hold zero
memory will speak crash.

Architecture Registers

Register Name	Syntax	Index	Purpose
Zero Pointer	$zp	000	This register always contains the value zero (0)
Instruction Pointer	$ip	001	This register points to the current instruction
Return Pointer	$rp	010	This register points to the instruction that should be returned to after function calls
Function Pointer	$fp	011	The register points to the start of a function to be called
Stack Pointer	$sp	100	The register that points to the top of the memory stack
Delta X	$dx	101	The value set to this register will be added to the X-Component of the satellite's position or orientation as specified. Expects a IEEE 754 half-precision floating point value. (some ancient 1985 standard
Delta Y	$dy	110	The value set to this register will be added to the Y-Component of the satellite's position or orientation as specified. Expects a IEEE 754 half-precision floating point value.
Delta Z	$dz	111	The value set to this reg... Z-Com...

Block Diagram - Reorientation



ALU_Ctrl operational values:

ADD	00
OR	01
SHIFT LEFT-LOGICAL	10
	11

ABS [illegible]

abs [illegible]

Biben[illegible] quisque egestas. Commodo odio aenean sed adip[illegible]stique. Netus et malesuada fames ac turpis egestas maec[illegible]msan sit amet nulla facilisi morbi tempus iaculis urna id. [illegible] fringilla urna porttitor rhoncus.

ADC [illegible]

adc

Lorem ipsu[illegible]ctetur adipiscing elit, sed do eiusmod tempor incididunt ut la[illegible] aliqua. Malesuada proin libero nunc consequat. Diam vulputate ut pharetra sit. Suspendisse sed nisi lacus sed viverra tellus. Erat nam at lectus urna duis convallis. Cursus eget nunc scelerisque viverra mauris in aliquam. Parturient montes nascetur ridiculus mus mauris. Maecenas pharetra convallis posuere morbi leo urna.

ADD : Add Two Registers

`add        $rd $rs1 $rs2 :`

Adds the two unsigned integers within the two specified source registe[illegible] $rs2), and stores the result within the specified destination regis[illegible]

ADDI: Add Immediate

`addi       $rd $rs imm :`

Adds the unsigned value within the [illegible] (imm), and stores its value within [illegible]

AND : Logical And

`and        $rd $rs1 [illegible]`

Viverra ipsum nunc ali[illegible] magna etiam tempor [illegible] Ultrices [illegible] Eleifend [illegible]

JRD : Jump

```
jrd
```

Maece[illegible]rbi leo urna. Eu mi bibendum neque egesta[illegible]
congue [illegible]hean sed adipiscing diam donec adipiscing
tristique [illegible] egestas maecenas pharetra convallis.

JRDI

```
jrdi
```

Maecena[illegible] urna. Eu mi bibendum neque egestas
congue [illegible]ean sed adipiscing diam donec adipiscing
tristique [illegible]s egestas maecenas pharetra convallis.

LID : Lorem ipsum [illegible]

```
lid          $rd $rs1 $rs2
```

Sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Malesuada proin libero nunc consequat. Diam vulputate ut pharetra sit. Suspendisse sed nisi lacus sed viverra tellus. Erat nam at lectus urna duis convallis. Cursus eget nunc scelerisque viverra mauris in aliquam. Parturient montes

LUI : Load Upper Immediate

```
lui          $rd imm:
```

Loads the immediate value (imm) into the first eight (8) most significant bits of the [illegible]egister ($rd)

lorem donec massa sapien faucibus et.
donec pretium. Sagittis purus sit amet volutpat
nulla malesuada pellentesque elit eget
as sed felis eget

ODUI : O

`odui`

In iaculis nu donec massa sapien faucibus et.
Phasellus fauc ec pretium. Sagittis purus sit amet volutpat
consequat mauris n nulla malesuada pellentes
elit eget gravida. Consequat id porta nibh venenatis cras sed felis eg

ORI : Logical Or Immediate

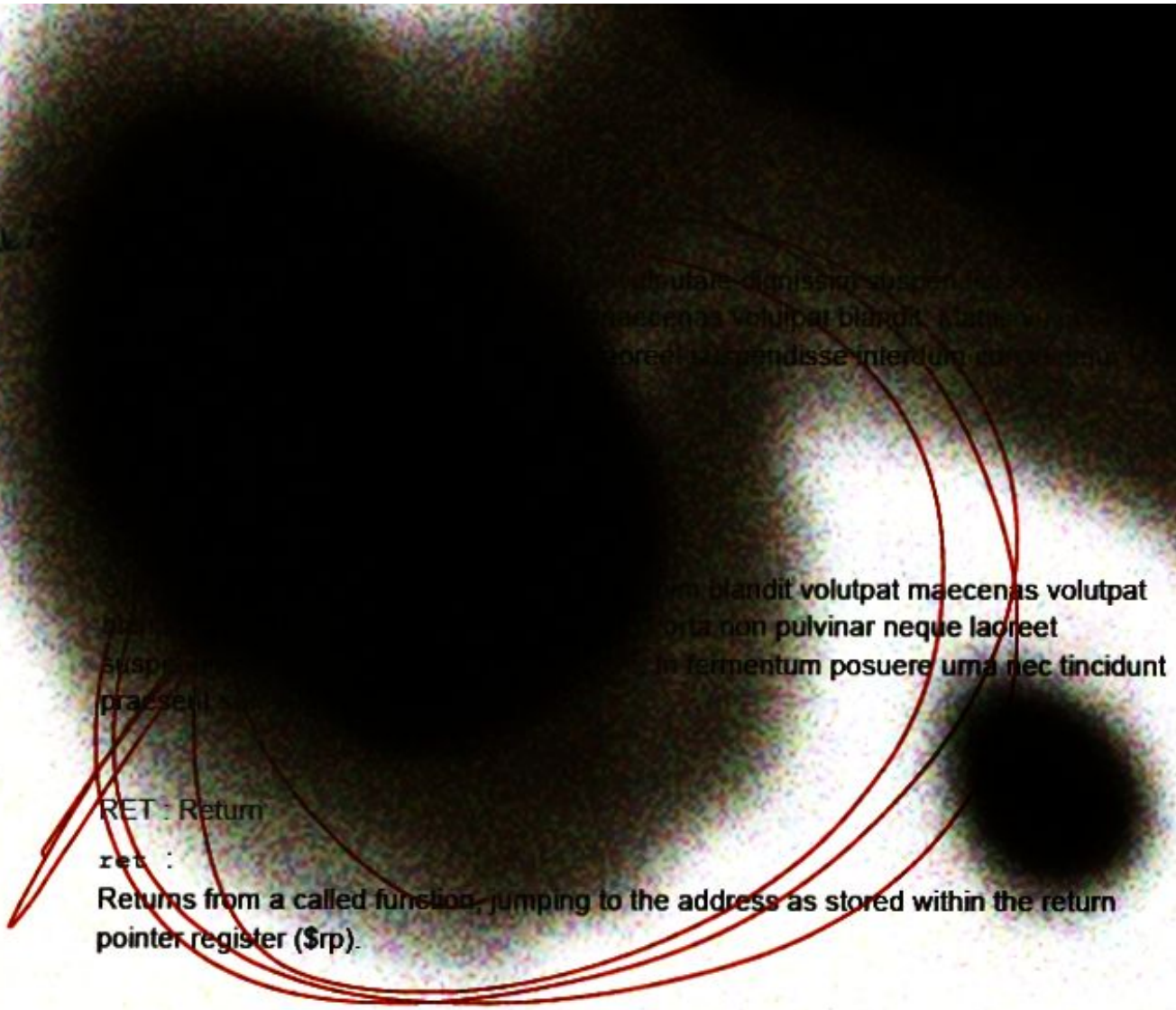
`ori $rd $rs imm` :

Performs a logical OR between the immediate value, (imm) value and the value within the selected source register ($rs), and stores the result within the selected destination register ($rd)

PMM : Pharetra Massa Massa

`pmm $rd $rs1 $rs2` :

ultricies mi. Malesuada pellentesque elit eget
vivamus at augue eget arc



RET : Return

`ret` :

Returns from a called function, jumping to the address as stored within the return pointer register ($rp).

SLLI : Shift-Left-Logical Immediate

`slli $rd $rs imm` :

Shifts the value within the specified source register ($rs) left by the number of bits specified with the immediate value (imm). The result of this value is stored within ($rd).