

---

---

# Blimp Instruction Set Architecture

---

---

---

## Instruction List

ECHO	<a href="#">2</a>
MOT—motor	<a href="#">3</a>
PING	<a href="#">4</a>
QRY—query	<a href="#">5</a>
RST—reset	<a href="#">6</a>
RUN	<a href="#">7</a>
STAT—status	<a href="#">8</a>
STOP	<a href="#">9</a>

---

# ECHO

**DESCRIPTION:** reply to a PING message

**COMMAND STRING:** \$ECHO;

<i>byte</i>	1	2	3	4	5	6
	\$	E	C	H	O	;

**LENGTH:** 6 bytes

**DIRECTION:** BLIMP → PC

**NOTES:**

The ECHO command takes no arguments. This is a response in reply to a PING message from the PC.

**EXAMPLE:**

\$ECHO;

---

# MOT—motor

**DESCRIPTION:** Low-level motor control commands  
**COMMAND STRING:** \$MOT{ID}{DIR}{SPEED};

<i>byte</i>	1	2	3	4	5	6	7	8	9	10
	\$	M	O	T	ID	DIR	SPEED			;

<b>ID</b>	Byte 5	ID of the motor to control
		0 Motor 0: left, port
		1 Motor 1: right, starboard
		2 Motor 2: up, down
<b>DIR</b>	Byte 6	Select the direction of the motor
		> Forward (motors 0/1)
		< Reverse (motors 0/1)
		v Down (motor 2 only)
		^ Up (motor 2 only)
<b>SPEED</b>	Bytes 7-9	Motor speed, as controlled by pwm duty cycle value
		000 0% duty cycle; minimum speed
		128 50% speed
		255 100% duty cycle; maximum speed

**LENGTH:** 10 bytes

**DIRECTION:** PC → BLIMP

## NOTES:

The low-level motor control commands control the speed of each of the three fans on the blimp.

It is an error to use v or ^ with a motor ID of 0 or 1. Likewise, > and < should only be used for motors 0 and 1 but not the up/down fan 2.

## EXAMPLE:

```
$MOT1>128;           # right fan forward at 50% speed
$MOT2v000;          # vertical fan down at 0% speed (stopped)
$MOT0>255;          # left fan forward at full speed
$MOT2^016;          # vertical fan up at speed 16
$MOT0<064;          # left motor reverse at speed 64
```

---

# PING

**DESCRIPTION:** keepalive or ping message

**COMMAND STRING:** \$PING;

<i>byte</i>	1	2	3	4	5	6
	\$	P	I	N	G	;

**LENGTH:** 6 bytes

**DIRECTION:** PC → BLIMP

**NOTES:**

The PING command takes no arguments. The blimp should send an ECHO command in response.

**EXAMPLE:**

\$PING;



---

# RST—reset

**DESCRIPTION:** Disable sensors and motors

**COMMAND STRING:** \$RST;

<i>byte</i>	1	2	3	4	5
	\$	R	S	T	\$

**LENGTH:** 5 bytes

**DIRECTION:** PC → BLIMP

**NOTES:**  
The RST command takes no arguments.

**EXAMPLE:**  
\$RST;

---

# RUN

**DESCRIPTION:** Enable sensors and motors

**COMMAND STRING:** \$RUN;

<i>byte</i>	1	2	3	4	5
	\$	R	U	N	;

**LENGTH:** 5 bytes

**DIRECTION:** PC → BLIMP

**NOTES:**

The run command takes no arguments.

**EXAMPLE:**

\$RUN;

---

# STAT—status

**DESCRIPTION:** Report the status of a subsystem  
**COMMAND STRING:** \$STAT{SYSID}{STATE};

<i>byte</i>	1	2	3	4	5	6	7	8
	\$	Q	R	Y	SYSID	STATE		;

**SYSID** Byte 5 ID of the system to query  
(see the QRY command for a list of possible SYSIDs)

**STATE** Bytes 6-7 State of the subsystem  
UP up, subsystem running normal  
DN down, subsystem not running  
ER error, subsystem error

**LENGTH:** 8 bytes

**DIRECTION:** BLIMP → PC

## NOTES:

The STAT command is usually in response to a QRY command, but it does not have to be. The blimp can send STAT commands to the PC at will if necessary.

## EXAMPLE:

```
$STATSUP;           # status of (main) system is up
$STATMER;          # motoion conroller has an error
$STATTDN;         # temperature subsystem is down
```



---

# STOP

**DESCRIPTION:** Disable sensors and motors

**COMMAND STRING:** \$STOP;

<i>byte</i>	1	2	3	4	5	6
	\$	S	T	O	P	;

**LENGTH:** 6 bytes

**DIRECTION:** PC → BLIMP

**NOTES:**  
The STOP command takes no arguments.

**EXAMPLE:**  
\$STOP;