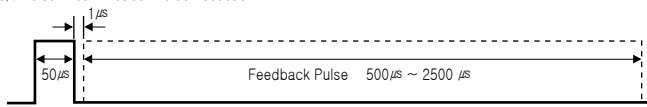


HMI (Hitec Multi-protocol Interface) provides you with an interface to program the HITEC Robot Servos. This information is only available for HITEC Robot Servos with firmware versions 1.10 or above. (To program all the features of the Robot Servos, an optional kit is required.)

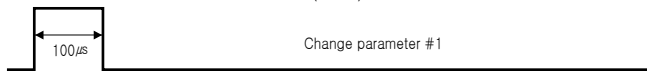
1. HMI Protocol Pulse

In this section, the structure of the HMI Protocol Pulse is described for advanced users wishing to know the exact positions of HITEC Robot Servos connected to a PC. Users can get the Feedback Data from servos using a Micom that is programmed to output 4 types of pulses to the servo and to receive a feedback pulses from the servo through the signal wire of the servos.

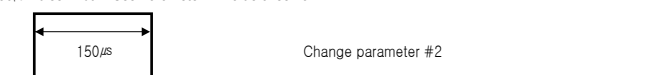
1) 50 μ s Pulse Width / Position Value Feedback



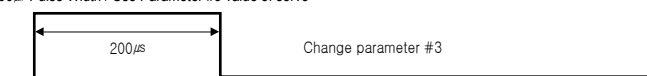
2) 100 μ s Pulse Width / Use Parameter #1 value of servo (default)



3) 150 μ s Pulse Width / Use Parameter #2 value of servo.



4) 200 μ s Pulse Width / Use Parameter #3 value of servo



- Parameter #1: Default
- Parameter #2 and #3: Reserved Parameter. It is possible to setup parameter #2 and #3 with the HMI Servo Programmer Kit (optional item). To set the servos operating characteristics during operation, a user will have to create a custom program using the open source code found at <http://www.hitecrobotics.com>.

Note1: To get position feedback using an external circuit, the communication port of the controller must be bi-directional, and the signal terminal should be setup in a Pull-Up state.

Note2: Because the positional feedback mentioned in note 1 operates in conjunction with the PWM control function, there is a chance that a communication error will occur 10% of the time.

2. Parameters that can be set using the HMI Servo Programmer kit

- D-gain Parameter
- Dead Zone Parameter
- P-gain Parameter

3. PC to Servo Serial Interface

- Through the serial port of a PC, HMI can control up to 127 HITEC Robot Servos (which are programmable with HMI) directly without any interconnection devices.

- The number of servos that can be controlled through the serial port is dependent on the PC environment and interface (especially on the signal level of the serial port).

- In the case of Voltage/Current Feedback, the value is presented as an integer (0~256), so users will have to convert the integer to make sense of it as a physical value.

1) B_Version : Verifying ID and Version <- Feedback the ID and Version of the Servo

B_Version	80	E7	0	0	version	ID
-----------	----	----	---	---	---------	----

>> Packet Command

HEADER	Command	Data1	Data2	CHKSUM	Receive[0x00]	Receive[0x00]
1byte	1byte	1byte	1byte	1byte	1byte	1byte

- header = 0x80 (packet start)
 - command = 0xE7
 - data1 = 0x00
 - data2 = 0x00
 - check = header + command + data1 + data2
 - NULL = 0x00
 - NULL = 0x00

>> Packet Receive

Recieve[Data1]	Receive[Data2]
1byte	1byte

- data1 = version <-version feedback
 - data2 = ID <-ID feedback

2) B_Battery : Verifying Voltage and Current.

B_Version	80	E8	0	0	current	voltage
-----------	----	----	---	---	---------	---------

>> Packet Command

HEADER	Command	Data1	Data2	CHKSUM	Receive[0x00]	Receive[0x00]
1byte	1byte	1byte	1byte	1byte	1byte	1byte

- header = 0x80(packet start)
 - command = 0xE8
 - data1 = 0x00
 - data2 = 0x00
 - check = header + command + data1 + data2
 - NULL = 0x00
 - NULL = 0x00

>> Packet Receive

Recieve[Data1]	Receive[Data2]
1byte	1byte

- data1 = current <-Current feedback
 - data2 = voltage <-Voltage feedback

3) B_ID_R_POS_PC : Setting up motor speed and verifying position

B_ID_R_POS_PC	80	E9	00-7F(ID)	speed	pos_H	pos_L	ID
---------------	----	----	-----------	-------	-------	-------	----

>>Packet Command

HEADER	Command	Data1	Data2	CHKSUM	Receive[0x00]	Receive[0x00]
1byte	1byte	1byte	1byte	1byte	1byte	1byte

- header = 0x80 (Packet Start)
 - command = 0xE9
 - data1 = 00-7F(ID)
 - data2 = speed <- Setting up speed
 - check = header + command + data1 + data2
 - NULL = 0x00
 - NULL = 0x00

>> Packet Receive

Recieve[Data1]	Receive[Data2]
1byte	1byte

- data1 = pos_H <- High byte of position feedback value
 - data2 = pos_L <- Low byte of position feedback value

4) B_ID_W_MOV_MAX : Setting up motor position

B_ID_W_MOV_MAX	80	E9	00-7F(ID)	pos_H	pos_L	-	ID
----------------	----	----	-----------	-------	-------	---	----

>> Packet Command

HEADER	Command	Data1	Data2	CHKSUM	Receive[0x00]	Receive[0x00]
1byte	1byte	1byte	1byte	1byte	1byte	1byte

- header = 0x80(Packet Start)
 - command = 00-7F(ID)
 - data1 = pos_H <- Writing high byte of position command
 - data2 = pos_L <- Writing low byte of position command
 - check = header + command + data1 + data2
 - NULL = 0x00
 - NULL = 0x00

>> Packet Receive

Recieve[Data1]	Receive[Data2]
1byte	1byte

- data1 = 0x00 <- High byte of current position value feedback
 - data2 = 0x00 <- Low byte of current position value feedback

5) B_motor_go_stop : Motor operation setup

B_motor_go_stop	80	EB	0	0/1	03	03	0:stop 1:go
-----------------	----	----	---	-----	----	----	-------------

>> Packet Command

HEADER	Command	Data1	Data2	CHKSUM	Receive[0x00]	Receive[0x00]
1byte	1byte	1byte	1byte	1byte	1byte	1byte

- header = 0x80 (Packet Start)
 - command = EB
 - data1 = 0
 - data2 = 0x00 / 0x01 (0-stop / 1-go)
 - check = header + command + data1 + data2
 - NULL = 0x00
 - NULL = 0x00

>> Packet Receive

Recieve[Data1]	Receive[Data2]
1byte	1byte

- data1 = 0x03
 - data2 = 0x03

You can find more detailed information at the HITEC Robotics Home Page (<http://www.hitecrobotics.com>) and download all articles of interest.

4. Overload Protection

The HITEC Robot Servos have an Overload Protection Function, which is intended to protect motor, amplifier, and Robot.

- If overload occurs, the overload protection is activated and the power to the servo is turned off within 10 seconds.
- Overload protection is removed when the power is cycled to the servo.
- For safety, the overload protection function is not user-programmable.

5. After the power to the servo has cycled, it will move slowly to neutral. Thereafter, it will operate normally. This is not a malfunction.

6. Robot Servos usually have high torque, so users need to be very careful handling the robot while it is operating. Carelessly touching a joint or servo may cause injury.

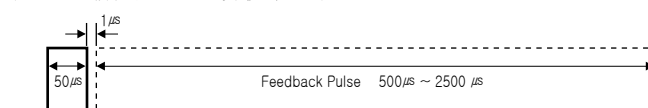
7. HITEC Robot Servos are developed for Robot Control only. Use of them for the other applications (RC, etc) is prohibited.

HMI (Hitec Multi-protocol Interface) は Hitec ロボットサーボシリーズの各種設定をユーザーが任意に変更可能にするものです。この機能説明書は、Hitec ロボットサーボ (HSR) シリーズで、ファームウェアバージョン 1.10 以上の機種に適用されます。(ロボットサーボの設定には、別途接続キット (オプション) が必要になります。)

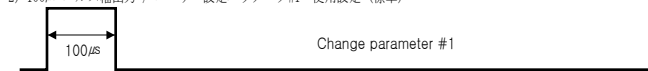
1. HMI 通信パルス設定

本項目では、HMI 機能を利用するために必要なパルス設定について説明します。下記 内容に基づき Hitec ロボットサーボの各種設定値を PC 等に取り込むことが出来ます。ユーザーが用意したマイコンボードから下記に示す 4 種類のパルスを出力することで、サーボモータの信号線から各種フィードバックを受ける事ができます。

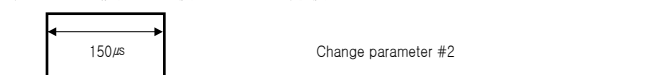
1) 50 μ s パルス幅出力 / サーボモータ現在値フィードバック



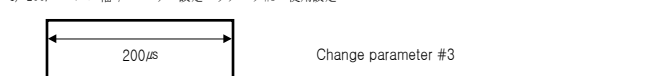
2) 100 μ s パルス幅出力 / ユーザー設定パラメータ#1 使用設定 (標準)



3) 150 μ s パルス幅 / ユーザー設定パラメータ#2 使用設定



4) 200 μ s パルス幅 / ユーザー設定パラメータ#3 使用設定



■ パラメータ #1: 標準設定 (工場出荷時)

- パラメータ #2 及び #3: 予備のパラメータです。パラメータ #2 及び #3 は、HMI サーボプログラマーキット (別売オプション) を利用することで設定が可能になります。但し、動作中のサーボモータのパラメータ設定を変更する場合は、別途公開されているソースコードを参考にしてユーザー側で設定プログラムを用意する必要があります。(参考: <http://www.hitecrobotics.com>)

注記 1) マイコン回路を利用して位置情報フィードバックを受け取るために、コントローラのポートは入出力双方向通信が可能なものを使用してください。また、入力信号はプルアップ抵抗の接続を行ってください。

注記 2) 上記ポジションフィードバックは、ラジコン用 PWM 制御に基づいて操作されるため、絶対的な位置に対して 10% 程度の誤差が生じることがあります。

2. HMI サーボプログラマーキットによって設定可能なパラメータ

- 比例ゲインパラメータ
- デッドゾーンパラメータ
- 位置ゲインパラメータ

3. PC - サーボ間 シリアルインターフェイス

- PC のシリアルポートを経由し、HMI インターフェイスを利用することで最大 127 個の Hitec ロボットサーボ (注: HMI プログラミング可能なモデルに限る) をコントロールすることが可能になります。

- シリアルポートを経由して制御可能なサーボの数は接続されるシリアルポートの環境 (特にシリアルポートの信号レベルが影響します) により前後します。

- 電圧 / 電流フィードバックを行う際、データの戻り値は整数 (0 ~ 256) で入力される為、使用環境の電源等に基づいた変換表をユーザー側で用意する必要があります。

1) B_Version : ID とバージョンの確認 <- サーボモータの ID とバージョンをフィードバックします

B_Version	80	E7	0	0	version	ID
-----------	----	----	---	---	---------	----

>> 送信コマンド

HEADER	Command	Data1	Data2	CHKSUM	Receive[0x00]	Receive[0x00]
1byte	1byte	1byte	1byte	1byte	1byte	1byte

- header = 0x80 (通信開始)
 - command = 0xE7
 - data1 = 0x00
 - data2 = 0x00
 - check = header + command + data1 + data2
 - NULL = 0x00
 - NULL = 0x00

>> 受信データ

Recieve[Data1]	Receive[Data2]
1byte	1byte

- data1 = version <-バージョンフィードバック
 - data2 = ID <-サーボ ID フィードバック

2) B_Battery : 電圧及び電流の確認

B_Version	80	E8	0	0	current	voltage
-----------	----	----	---	---	---------	---------

>> 送信コマンド

HEADER	Command	Data1	Data2	CHKSUM	Receive[0x00]	Receive[0x00]
1byte	1byte	1byte	1byte	1byte	1byte	1byte

- header = 0x80 (通信開始)
 - command = 0xE8
 - data1 = 0x00
 - data2 = 0x00
 - check = header + command + data1 + data2
 - NULL = 0x00
 - NULL = 0x00