CH-1092 Belmont, Switzerland
info@didel.com, www.didel.com

HESO Format (Hexa Simplified with Orders)

HESO is a simple transfer format which allows a master (usually a PC) to send orders and transfer data to and from one or several microcontrollers, over a 3-wire RS232 line.

The protocol principle is simple and uses only standard Ascii characters that will be transmitted by any terminal simulator and communication driver. A command consits of an 8-bit parameter and an order letter. The parameter is sent as zero, one or two "HES" caracters (see below). The order is an upper case letter. The slave reply with an 8-bit parameter in the same format, followed by the corresponding lower case letter.

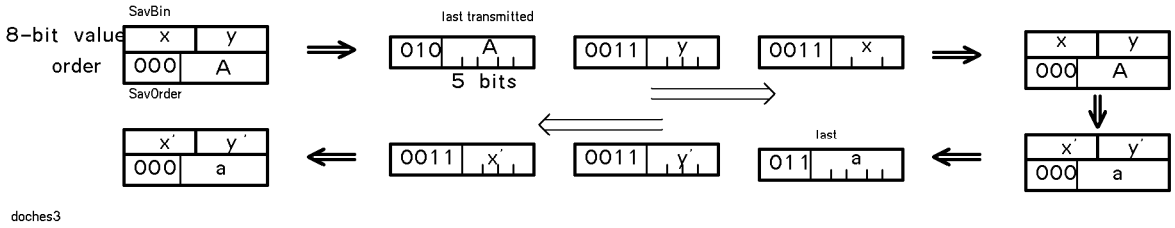
The 8-bit parameters are transmitted as 2 nibbles, converted in Ascii by adding the Ascii code of digit zero. The 0 to 9 nibbles are transferred as usual, but hexa nibbles higher than 9 are not converted into lettres A .. F, but transmitted as the Ascii caracters following the 9, that is : ; > = < ? (HES code) for hex A to F. This simplifies the conversion at both sides and leaves the aplhabet free for orders. If more than two digits are transmitted, only the last two are kept. This facilitates the correction of typing errors when using a terminal, since the delete code is not available.

Of course, one can use as a subset of the HES codes the digits 0 to 9, and consider HES numbers as BCD numbers. The value range is reduced to 00-99 instead of 00-??, that is 0 to 256, but the legibility is considerably increased. The routine for converting the e.g. 0-255 values of an A/D converter into to good approximation 00-99 equivalents can be found in www.didel.com/doc/DopiHeso.pdf.

Orders are coded as upper case letters, codes 16´40 (0x40 or 040h if you prefer) to 16´5E (signs @ A B ...] ^), 31 different orders that can be associated with an optional preceeding 8-bit parameter. Hence the name HESO for the protocol: HExa Simplified with Orders.

The slave replies, when the order is executed, by an optional parameter with the same format, followed by the corresponding lower case letter (codes 16´60 to 16´7E). Code 16´7F is not used since it correspond to the Delete character, and is handled in different ways by terminals.

The Ascii dump of an HESO transaction is easy to decode, and facilitates the debugging.



As an oversimplified example, the slave may be a motor moving an arm, controlled by a microcontroller accepting the following orders:

A - Motor moves to an initial position
nnB - Starts motor at speed nn
C - Give the position of the arm actuated by the motor

A typical exchange, for asking the system to go in a wanted position, is given below.

Master order	Slave reply	Slave action
A		Move arm in initial position
	a	(reply when it is done)
5;B		Start the motor with speed 16´5B=2´010101011
	b	(reply immediately)
C		Read and send position of the arm (=16´1F=2´00011111)
	1?c	(reply immediately)

- The master can continue to send C orders until the position is reached.

Multitasking

Orders may take some time to be executed, and the microcontroller does not handle a queue of incoming command. The slave can be programmed to accept different orders and acknowledge them when they are finished. The constraint is to avoid sending a command to the same order until it is finished. An overlapping order will be ignored (underline code _ is reserved for possible future error signalling). It is indeed preferable, and required for multislave operations, to define only immediate reply orders, and add "test ready" orders, to be repeated until slave is ready.

Multiple slaves

Multiples slaves are possible: order 16´40 (letter @) selects the slave. The address corresponds to the parameter in front of the @ (@ alone, that is 00@ is reserved for possible broadcast to all slaves or for a general software reset). A slave, which has recognized its address accepts the orders until an @ address selects another unit. On the master, a timeout must take care of non-existant units selected by mistake.

Hence, there are 255 possible slaves (electrical limitation to less than 10) which can receive 31 orders, the set of orders being of course specific to the slave. Each transaction transfers one data byte of information in both direction. Data bytes are optional, but an non existant byte cannot be distinguished from a byte with value 0.

Microcontroller routines

On the microcontroller, the routine to send and receive a byte are quite simple. The PIC routines are given in Appendix and can be accessed at www.didel.com/doc//DopiHeso.html.html.

Master routine

The procedures to send and receive an Heso order in C are trivial. 48 is the Ascii code of "0" and 64 the code of "@".

```
void SndHeso (char SavBin, char SavOrder)
      {
      char code ;                    //local variable
      code = (SavBin >> 4) + 48 ;    // High nibble
      Sendcar (code) ;
      code = SavBin & 0x0F + 48 ;   //Low nibble
      Sendcar (code) ;
      code = SavOrder + 64 ;
      Sendcar (code) ;
      }
```

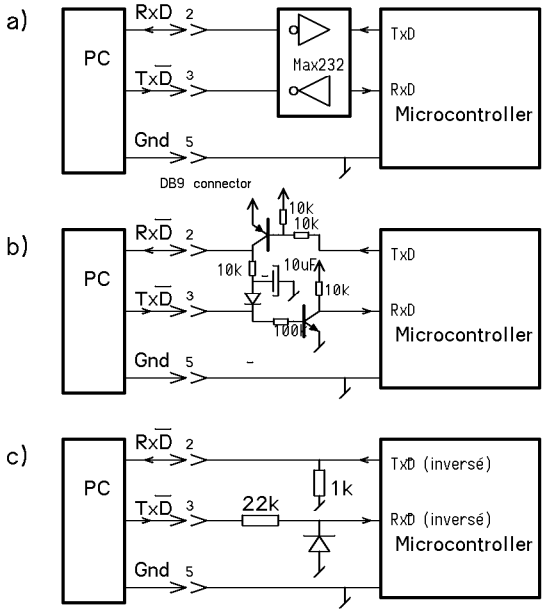
```
void  RecHeso (char SavBin, char SavOrder)
      {
      char code ;
      SavBin = 0 ;
      Getcar (code) ;
      while (code >= 48) ;
       {
           SavBin = SavBin << 4 ;
           SavBin = SavBin + code & 0x0F ;   // get 4 l
       }
      SavOrder= code/64
```

Slave routines

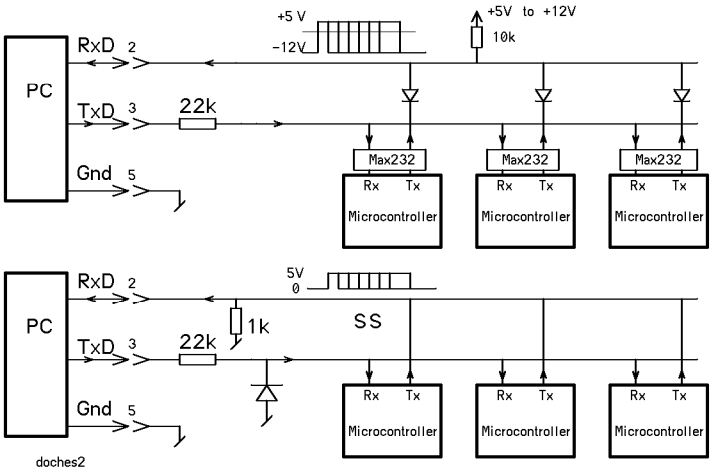
PIC serial send and receive routines are documented on the document www.didel.com/doc/DopicSer.pdf and HESO routines on www.didel.com/doc/DopiHeso.pdf.

Configurations

The simplest configuration is when a master controls a single slave. If the slave is RS232 compatible, they are just connected together. Usually, the slave will be a quite simple microcontroler, and there are three basic connection schemes. On figure 2a, a Max 232 or equivalent circuit handles the interface. Cheaper, but out of date, is the schematic of figure 2c. The cheapest solution uses the 3 components of figure 2d, and is compatible with most PC (levels are 0 to 5V and not +5/-5 minimum), but the software must handle the inversion inside the processor; this may be incompatible with the microcontroller serial hardware.



Multimaster configurations need OR gates to send the information back to the master. These gates can be implemented with diodes or open collector outputs. Direct connection of the microcontroller is possible, but the software must invert the TxD/RxD signals. Units are selected with a nn@ command prior to the order. Multitasking with delayed replies is not possible since asynchronous replies could overlap.



Code Ascii

Binaire		0000	0001	0010	0011	0100	0101	0110	0111
	Hexa	0	10	20	30	40	50	60	70
0000	0	NUL	DLE	space	0	@	P	`	p
0001	1	SOH	DC1	!	1	A	Q	a	q
0010	2	STX	DC2	"	2	B	R	b	r
0011	3	ETX	DC3	#	3	C	S	c	s
0100	4	EOT	DC4	$	4	D	T	d	t
0101	5	ENQ	NAK	%	5	E	U	e	t
0110	6	ACK	SYN	&	6	F	V	f	u
0111	7	BEL	ETB	´	7	G	W	g	v
1000	8	BS	CAN	(	8	H	X	h	x
1001	9	HT	EM	)	9	I	Y	i	y
1010	A	LF	SUB	*	:	J	Z	j	z
1011	B	VT	ESC	+	;	K	[	k	{
1100	C	FF	FS	,	<	L	\	l	|
1101	D	CR	GS	-	=	M	]	m	}
1110	E	SO	RS	.	>	N	^	n	~
1111	F	SI	US	/	?	O	_	o	DEL