DIGITAL LOADCELL

MODEL 740D 15 ... 60t

USER SPECIFICATION

INDEX

1 GENERAL DESCRIPTION FOR DIGITAL LOAD CELL 2

2 SERIAL INTERFACE RS-485 (4 WIRE, FULL DUPLEX) 3

2.1 CHARACTERISTICS OF THE INTERFACE 3
2.2 ELECTRICAL CONNECTIONS 3
2.3 COMMUNICATIONS 4

3 COMMANDS 5

3.1 COMMAND FORMAT. 5
3.2 ANSWERS TO COMMANDS. 5
3.3 OUTPUT FOR THE MEASURED VALUE 6
3.4 COMMANDS OVERVIEW: 6

4 COMMANDS DESCRIPTION 7

4.1 ADR. SET BUS ADDRESS. 7
4.2 FIL. SET FILTER 8
4.3 VAL. READ WEIGHT 9
4.4 ZER. CALIBRATE USER ZERO 10
4.5 GAI. CALIBRATE USER GAIN 11
4.6 NOM. SET NOMINAL SCALING 12
4.7 RES. DEVICE RESET 13
4.8 RDV. RELOAD DEFAULT VALUES 14
4.9 VER. SW VERSION 15
4.10 TRG. WEIGHT TRIGGER 16
4.11 STU. LOAD CELL STATUS 17
4.12 BAU. CHANGE BAUD RATE 18
4.13 CAP.NOMINAL LOADCELL CAPACITY 19
4.14 CHK.CHECKSUM FOR TRANSMITTED WEIGHT 20

5 REVISION HISTORY 21

6 APENDIX A CHECKSUM ALGORITHMS 22

1 General description for digital load cell

- Operating voltage: 7,5 ... 18 VDC
- Serial interface RS-485 (4 wire, full-duplex)
- Digital filter scalable
- Storage of parameters in non-volatile memory
- Internal resolution: 24 bit, sigma-delta AD converter, ± 3mV/V
- All settings through serial interface
- SW updateable through serial interface
- Lightning protection

2 Serial Interface RS-485 (4 wire, full duplex)

Up to 32 digital load cells can be connected to a common bus line through the RS-485 interface. The bus mode is designed in a master-slave configuration, whereby the digital load cells are implemented as slaves. So, all system activities are initiated by the host computer or the digital weight indicator.

Each load cell receives its own communication address (01 - 32). A broadcast command (device address 00) can be used to send some special commands to all slave devices at the same time. In that case no slave will send acknowledge.

Cable lengths up to 1000 m can be achieved with the RS-485 bus. For EMC reasons a shielded cable with twisted pair configuration is recommended. Since the RS-485 bus is a differential bus interface, the signal levels are states of differential voltage between the lines and are not related to ground.

Termination resistors for the electrical function of the bus system are needed at the ends of the bus lines.

2.1 Characteristics of the interface

Start bit:	1
Data length:	8 bits
Stop bit:	1
Parity:	no parity
Handshake:	not implemented
Baud rate:	from 4800 to 38400 baud

2.2 Electrical connections

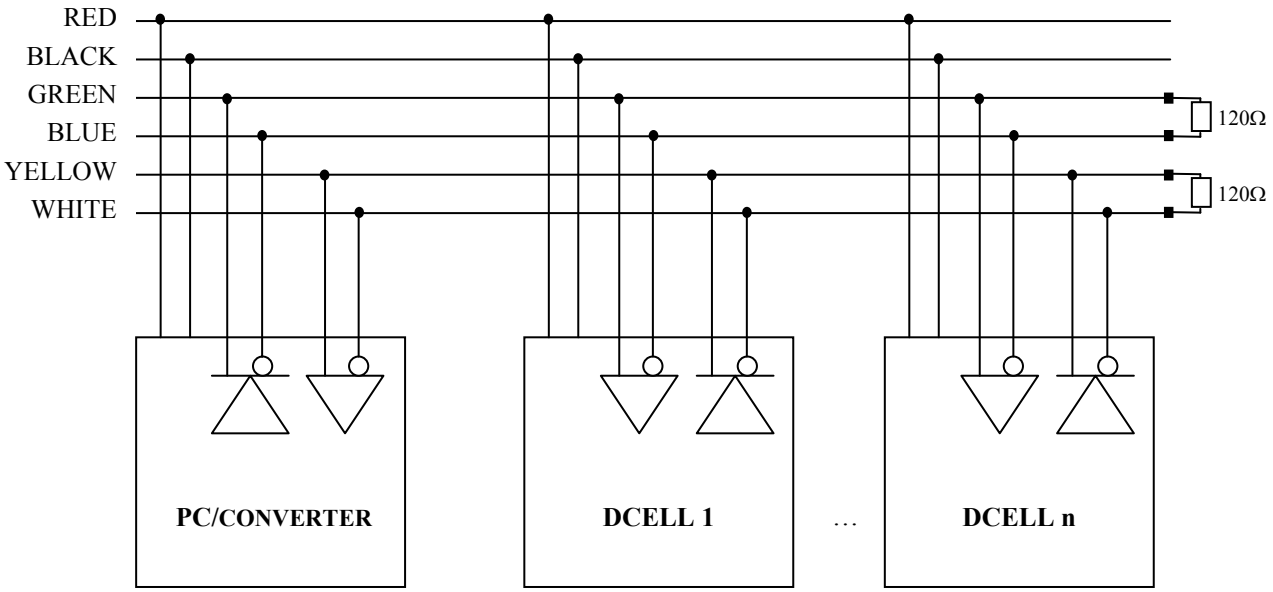
Cable code and colour:

Red:	U_B +7,5 ... +18V DC
Black:	GND
Green:	RB (+)
Blue:	RA (-)
Yellow:	TB (+)
White:	TA (-)
Shield:	twisted, tinned

External Trigger Input: NOT IMPLEMENTED IN THIS VERSION

2.3 Communications

Next figure is an overview of the bus connections. Two 120Ω resistors are needed at the end of the bus to match the line impedance. You can connect up to 32 digital cells on an RS-485 bus.



3 Commands.

The commands can be classified into:

- Interface commands (e.g.: ADR)
- Commands for adjusting and scaling (e.g.: ZER, GAI, NOM)
- Commands for the measuring mode (e.g.: FIL, VAL)
- Special commands (e.g.: RES, RDV)

3.1 Command format.

The commands have to be input in ASCII and uppercase.

Each command consists of the command initials, the device address, the parameters and the termination character (carriage return 0DH). Only if the termination character has been sent, the slave device will clear the input buffer and will be able to accept the next command.

The parameters specified with rounded brackets "(param.)" in the commands description are necessary and should be entered. Parameters in squared brackets "[param.]" are optional. The brackets themselves are not entered.

Whatever parameter surrounded by “less than” and “more than” symbols "<param.>" are ASCII non-printable characters and they are represented by its hexadecimal number.

With numeric entries, leading zeros are suppressed, numbers must be entered in direct format. For negative values first enter de negative sign (-) and then the number.

Example: ZER24,-452<CR> Sets the user zero to -452 counts.

3.2 Answers to commands.

Answers to inputs:

	Answer	End character
Correct input	ACK (ASCII 0x06)	<CR>
Incorrect input	NAK (ASCII 0x15)	<CR>

Exceptions: The command VAL returns the measured value.
Broadcast commands do not return answers.

3.3 Output for the measured value.

The output of the measured signal depends on the checksum configuration of the loadcell and will be send in ASCII.

Output without checksum:

S	**W**	**W**	**W**	**W**	**W**	**W**	**W**	**CR**

Output with checksum:

S	**W**	**W**	**W**	**W**	**W**	**W**	**W**	**CH**	**CL**	**CR**

S - sign: SPACE for measured value > 0; - for < 0
W - weight: 7 digits, MSB first
CH, CL - checksum 2 caracters ASCII
CR - termination (carriage return 0x0D)
Range: - ± 9999999

3.4 Commands Overview:

Command	*Function*	*See explanation in*
ADR	Set Bus Address	Section 4.1
FIL	Set Filter	Section 4.2
VAL	Read Weight	Section 4.3
ZER	Calibrate User Zero	Section 4.4
GAI	Calibrate User Gain	Section 4.5
NOM	Set Nominal Scaling	Section 4.6
RES	Device Reset	Section 4.7
RDV	Reload Default Values	Section 4.8
VER	SW Version	Section 4.9
TRG	Weight Trigger	Section 4.10
STU	Load Cell Status	Section 4.11
BAU	Change Baud Rate	Section 4.12
CAP	Read load cell nominal capacity	Section 4.13
CHK	Checksum	Section 4.14

4 Commands Description

4.1 ADR. Set Bus Address.

Range: 00 – 32, 99
Factory setting: 00
Parameters: 1/2

Entry: **ADR(device address),(new address)[,Serial No.]<CR>**
Entry of the device address as decimal number 00 - 32.

The serial number can be transmitted as a 3rd parameter. In this case the new device address is programmed only for the digital load cell with the serial number entered. Thus, it is possible to change device addresses of several load cells with the same programmed address (initialisation of the bus mode). Using device address 99 will change all load cells with the device address 00 to the new address and causes returning ACK<CR> from the load cell

Example 1: ADR01,25<CR>
The load cell with device address 01 will change to device address 25.

Example 2: ADR00,25,456789<CR>
The load cell with serial number 456789 will change its device address to 25. In this case the load cell with the serial number 456789 will return ACK<CR>.

Example 3: <u>Special case device address 99:</u>
ADR99,25<CR>
All load cells with the device address 00 (factory default) will change its device address to 25.

Returns: ACK<CR>

Query: **ADR(device address)?<CR>**
Returns the load cell serial number as a decimal number and its device address.

Example: ADR25?<CR>

Returns: 11 characters<CR> (e.g.: 00456789:25<CR>)

4.2 FIL. Set Filter

Range: 0 - 6
Factory setting: 4
Parameters: 1

Entry: **FIL(device address),(filter parameter)<CR>**
Entry of the filter parameter for the load cell with this device address.

Example: FIL25,6<CR>
Changes the filter parameter to 6 for the load cell with device address 25

Returns: ACK<CR>

Query: **FIL(device address)?<CR>**
Returns the filter parameter for the load cell with this device address.

Example: FIL25?<CR>

Returns: 11 characters<CR> (e.g.: 00000006:25<CR>)

4.3 VAL. Read Weight

Range: ----
Factory setting: ----
Parameters: ----

Entry: **VAL(device address)<CR>**
Returns the measured value (see format in section 3.3).

Example: VAL25<CR>
Load cell with device address 25 sends the measured value to the host

Returns: 8 characters<CR> (e.g.: -0052514<CR>)

Query: **----**

New for software version 1.009:
In the case of ADC Fault no weight will be transmitted for VAL command.

4.4 ZER. Calibrate User Zero

Range: ± Nominal Scaling (See NOM command)
Factory setting: 0
Parameters: 1

Entry: **ZER(device address)[, Zero Value]<CR>**
Adjustment of the zero of the load cell. This function can be made by using automatic calibration of the load cell electronics or by direct entry of the zero value.

Example1: ZER25<CR>
The load cell electronic measures the input signal and stores this value as the user zero.

Example2: ZER25,-0034567<CR>
Instead of measuring, the zero is entered and stored directly.

Returns: ACK<CR>

Query: **ZER(device address)?<CR>**
Returns the zero value

Example: ZER25?<CR>

Returns: 11 characters<CR> (e.g.: 00000014:25<CR>)

4.5 GAI. Calibrate User Gain

Range: ± 9.999999 (zero is not permitted)
Factory setting: 1.000000
Parameters: 1

Entry: **GAI(device address), (User Gain)<CR>**
Adjustment of the user gain of the load cell.

Format:

9.999999	(8 chars) (positive number)
+9.999999	(9 chars) (positive number)
<SPACE>9.999999	(9 chars) (positive number)
-9.999999	(9 chars) (negative number)

Example: GAI25,1.000050<CR>
The gain value is entered and stored without sign.

Returns: ACK<CR>

Query: **GAI(device address)?<CR>**
Returns the user gain value.

Example: GAI25?<CR>

Returns: If positive: 11 characters<CR>
(e.g.: 1.000000:25<CR>)
If negative: 12 characters<CR>
(e.g.: -1.000000:25<CR>)

4.6 NOM. Set Nominal Scaling

Range: 1 - 1000000
Factory setting: 200000
Parameters: 1

Entry: **NOM(device address),(Nominal scaling)<CR>**
Sets the output value for nominal load on the load cell.

Example: NOM25,0250000<CR>
The nominal scaling is entered and stored to 250000.

Returns: ACK<CR>

Query: **NOM(device address)?<CR>**
Returns the nominal scaling

Example: NOM25?<CR>

Returns: 11 characters<CR> (e.g.: 00200000:25<CR>)

4.7 RES. Device Reset

Range: ----
Factory setting: ----
Parameters: ----

Entry: **RES(device address)<CR>**
The command RES resets the device.

Example1: RES25<CR>
The load cell with the device address 25 will be reseted in 100ms after sending ACK.

Example2: RES00<CR>
All load cells connected to the bus will be reseted.

Returns: ACK<CR>
CAUTION: Address 00 (Broadcast) returns NOTHING

Query: **----**

4.8 RDV. Reload Default Values

Range: ----
Factory setting: ----
Parameters: ----

Entry: **RDV(device address)<CR>**
The command RDV sets and stores all parameters to the factory values. After that it resets the device.

CAUTION: This function also will set the device address to 00, so it will be necessary to configure it again.

Example1: RDV00<CR>
All load cells connected to the bus will execute this command

Example2: RDV25<CR>
The load cell with the device address 25 will set and store the factory default values and then will reset.

Returns: ACK<CR>
(CAUTION: Address 00 (Broadcast) returns NOTHING)

Query: **----**

4.9 VER. SW Version

Range: ----
Factory setting: ----
Parameters: ----

Entry: **----**

Query: **VER(device address)?<CR>**
The command VER returns the software version running on the load cell with this particular device address.

Example: VER25?<CR>
The load cell with the device address 25 will send its software version.

Returns: 9 characters<CR> (e.g.: 01.003:25)

4.10 TRG. Weight Trigger

Range: ----
Factory setting: ----
Parameters: ----

Entry: **TRG(device address)<CR>**
Save the current weight value to an internal memory space. This value will be returned when a trigger query is sent.

Example: TRG25<CR>
Load cell with device address 25 saves the measured value to an internal memory.

Returns: ACK<CR>
(CAUTION: Address 00 (Broadcast) returns NOTHING)

Query: **TRG(device address)?<CR>**
Return the value stored in the internal memory space.

Example: TRG25?<CR>
Load cell with device address 25 returns the stored weight value.

Returns: 8 characters<CR> (e.g.: -0068377<CR>)

New for software version 1.009:
In the case of ADC Fault no weight will be transmitted for TRG command.

4.11 STU. Load Cell Status

Range: ----
Factory setting: ----
Parameters: ----

Entry: **----**

Query: **STU(device address)?<CR>**
Return several bits (TRUE (1) or FALSE (0)).
The meanings of each bit are the following:

- Bit 0: Non-Volatile Memory corrupted
- Bit 1: ADC Fault (ADC doesn't respond)
- Bit 2: Weight-Reading Error (ADC returns an error)
- Bit 3: Reserved
- Bit 4: Reserved
- Bit 5: Reserved

(bit 0 is the FIRST char transmitted)

This command is useful to ensure that the data read from the load cell is correct, and that no error in measurement has been made. It could be convenient to be queried at fixed intervals of time (5 or 10 minutes, for example).

This option is useful due to the fact that the slave device cannot transmit any data to the master device in case of an internal malfunction. Master must ask the slave its internal state.

Example: STU25?<CR>
Load cell with device address 25 returns its internal state. For example: "001000"<CR>, meaning that nothing is connected to the ADC input.

Returns: 6 characters<CR> (e.g.: 001000<CR>)

New for software version 1.009:
In the case of ADC Fault no weight will be transmitted for VAL and TRG commands.

4.12 BAU. Change Baud Rate

Range: 4800bauds, 9600bauds, 19200bauds, 38400bauds
Factory setting: 19200bauds
Parameters: 1

Entry: **BAU(device address), (Baud Rate)<CR>**
Change communications baud rate.
The configuration options are 4800, 9600, 19200 and 38400.
All other baud rates can not be used.

CAUTION: If this command is correctly received by the load cell, it returns an ACK using the **OLD** baud rate. After that, the next communication will be performed using the **NEW** baud rate. If this command is NOT correctly received by the load cell, it returns a <NACK> and baud rate remains unchanged.

Example: BAU25,38400<CR>
Baud rate is changed to 38400bps.

Returns: ACK<CR>

Query: **BAU(device address)?<CR>**
Returns the current baud rate.

Example: BAU25?<CR>

Returns: 11 characters<CR> (e.g.: 00038400:25<CR>)

4.13 CAP.Nominal Loadcell Capacity

Range: ----
Factory setting: ----
Parameters: ----

Entry: **----**

Query: **CAP(device address)?<CR>**
Returns the nominal loadcell capacity in kg with 1 decimal

This command can be used during scale definition and calibration without test weights.

Example: CAP25?<CR>
Load cell with device address 25 returns its nominal capacity.
For example: "0030000.0:25"<CR>, for a 30000 kg loadcell.

Returns: 12 characters<CR> (e.g.: 0030000.0:25<CR>)

4.14 CHK.CheckSum for transmitted weight

Range: 0, 1, 2
Factory setting: 0 (OFF)
Parameters: 1

Entry: **CHK(device address), (parameter)<CR>**
Activates or disables checksum for transmitted weight.
Example1: CHK25,1<CR>
The loadcell with the bus-address 25 responds for VAL and TRG commands with transmitting the weight with XOR-checksum.
Returns: ACK<CR>

Example2: CHK25,2<CR>
The loadcell wth the bus-address 25 responds for VAL and TRG commands with transmitting theweight with CRC8-checksum
Returns: ACK<CR>

Query: **CHK(device address)?<CR>**
Returns the nominal loadcell capacity in kg with 1 decimal

Example: CHK25?<CR>
Returns: 11 characters<CR> (e.g.: 00000001:25<CR>)

The checksum parameter will not be saved in NVM, after every reset or power up the CHK parameter will be 0 (OFF). User must activate this function if desired.

5 Revision History

Rev.	Date	Modifications	SW Version
0	12-05-2004	Original document	1.002
1	24-05-2004	Added 2.3 topic	1.002
2	14-06-2004	Added command: *BAU*. Modified commands: *GAI* (zero gain is not enabled, new range ±9.999999), *ZER* (new range ±Nominal Scaling, user enters and reads this zero in "user units")	1.003
3	30-11-2005	max. operating voltage changed to 18V Added commands: - CAP sends nominal capacity of loadcell Modified commands: - ADR with device address '99' forces all loadcells with address '00' to new address - RES generates ACK if not broadcast	1.007
4	03-05-2007	Added command: - CHK calculates checksum for weight Modified commands: - VAL: in case of ADC_Fault no weight will be transmitted - TRG: in case of ADC_Fault no weight will be transmitted	1.009

6 APENDIX A Checksum algorithms

1. XOR - Checksum:
The first checksum we propose is a simple checksum – algorithm which uses XOR.

for example, if transmitted weight without checksum is:

	1	2	3	4	5	6	7	CR

	Hex	Bin	XOR
Space	20	0010 0000	
1	31	0011 0001	0001 0001
2	32	0011 0010	0010 0011
3	33	0011 0011	0001 0000
4	34	0011 0100	0010 0100
5	35	0011 0101	0001 0001
6	36	0011 0110	0010 0111
7	37	0011 0111	0001 0000

So the result of the sum of all XORs is 0001 0000 = 10hex.
This checksum is added (in ASCII) at the end of the value, so the
Transmitted weight with checksum is:

	1	2	3	4	5	6	7	1	0	CR

- Function for XOR checksum programmed in Visual Basic -

```
Private Function CalcXor(ByRef data() As Byte, nbytes As Integer) As Byte
'-------------------------------------------------------------------------
' Function:     (Private) CalcXor
' Description:  computes a 1 byte checksum using bytes 1 to nbytes of
'               the data to be transmitted
'
' Parameter:    Data ()As Byte      | data
'               nbytes as interger  | number of bytes in data
' Return Value: ClacXOR as Byte     | XOR value
'-------------------------------------------------------------------------
Dim x As Integer
Dim Checksum As Byte

Checksum = 0
For x = 1 To nbytes
    Checksum = data(x) Xor Checksum
Next x
CalcXor = Checksum

End Function
```

2. CRC8 - Checksum:

The CRC8, based on the polynomial expression: CRC8 = X^8 + X^5 + X^4 + 1 is a well known and often used standard to generate checksum for data transmission with more reliability like the simple XOR checksum. The CRC8 also gives the possibility to correct bit errors in data blocks. We want to avoid entering depth in the mathematical base of the calculation of the CRC8 checksum, but we can give you an example for a function to calculate the CRC8 checksum programmed in VisualBasic.

- Function for CRC8 checksum in VisualBasic -

```
Private Function CalcCRC8(ByRef data() As Byte, nbytes As Integer) As Byte
'-------------------------------------------------------------------------
' Function:     (Private) CRC8
' Description:  computes a 1 byte checksum using bytes 1 to nbytes of
'               the data to be transmitted
'
' Parameter:    Data ()As Byte      | data
'               nbytes as interger  | number of bytes in data
' Return Value: CalcCRC8 as Byte    | CRC value
'-------------------------------------------------------------------------
Dim crc, Index As Byte
Dim CRCtable1() As Variant
Dim x As Integer

crc = 0
CRCtable1 = Array( _
&H00, &H07, &H0E, &H09, &H1C, &H1B, &H12, &H15, _
&H38, &H3F, &H36, &H31, &H24, &H23, &H2A, &H2D, _
&H70, &H77, &H7E, &H79, &H6C, &H6B, &H62, &H65, _
&H48, &H4F, &H46, &H41, &H54, &H53, &H5A, &H5D, _
&HE0, &HE7, &HEE, &HE9, &HFC, &HFB, &HF2, &HF5, _
&HD8, &HDF, &HD6, &HD1, &HC4, &HC3, &HCA, &HCD, _
&H90, &H97, &H9E, &H99, &H8C, &H8B, &H82, &H85, _
&HA8, &HAF, &HA6, &HA1, &HB4, &HB3, &HBA, &HBD, _
&HC7, &HC0, &HC9, &HCE, &HDB, &HDC, &HD5, &HD2, _
&HFF, &HF8, &HF1, &HF6, &HE3, &HE4, &HED, &HEA, _
&HB7, &HB0, &HB9, &HBE, &HAB, &HAC, &HA5, &HA2, _
&H8F, &H88, &H81, &H86, &H93, &H94, &H9D, &H9A, _
&H27, &H20, &H29, &H2E, &H3B, &H3C, &H35, &H32, _
&H1F, &H18, &H11, &H16, &H03, &H04, &H0D, &H0A, _
&H57, &H50, &H59, &H5E, &H4B, &H4C, &H45, &H42, _
&H6F, &H68, &H61, &H66, &H73, &H74, &H7D, &H7A, _
&H89, &H8E, &H87, &H80, &H95, &H92, &H9B, &H9C, _
&HB1, &HB6, &HBF, &HB8, &HAD, &HAA, &HA3, &HA4, _
&HF9, &HFE, &HF7, &HF0, &HE5, &HE2, &HEB, &HEC, _
&HC1, &HC6, &HCF, &HC8, &HDD, &HDA, &HD3, &HD4, _
&H69, &H6E, &H67, &H60, &H75, &H72, &H7B, &H7C, _
&H51, &H56, &H5F, &H58, &H4D, &H4A, &H43, &H44, _
&H19, &H1E, &H17, &H10, &H05, &H02, &H0B, &H0C, _
&H21, &H26, &H2F, &H28, &H3D, &H3A, &H33, &H34, _
&H4E, &H49, &H40, &H47, &H52, &H55, &H5C, &H5B, _
&H76, &H71, &H78, &H7F, &H6A, &H6D, &H64, &H63, _
&H3E, &H39, &H30, &H37, &H22, &H25, &H2C, &H2B, _
&H06, &H01, &H08, &H0F, &H1A, &H1D, &H14, &H13, _
&HAE, &HA9, &HA0, &HA7, &HB2, &HB5, &HBC, &HBB, _
&H96, &H91, &H98, &H9F, &H8A, &H8D, &H84, &H83, _
&HDE, &HD9, &HD0, &HD7, &HC2, &HC5, &HCC, &HCB, _
&HE6, &HE1, &HE8, &HEF, &HFA, &HFD, &HF4, &HF3 )

For x = 1 To nbytes
    Index = crc Xor data(x)
    crc = CRCtable1(Index)
Next x
CalcCRC8 = crc

End Function
```