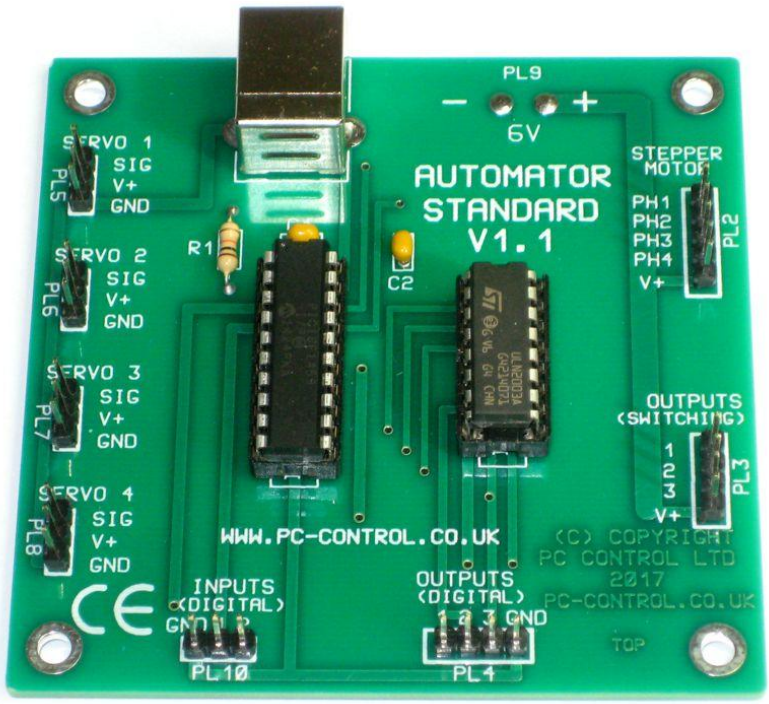
Automator
(Standard)

DLL Users Guide



Available exclusively from
PC Control Ltd.
www.pc-control.co.uk

Revision 1.2

Contents

1. Introduction

The Automator has been designed to be as easy to use as possible which is why it can be programmed by "non programmers" using a simple text file. If you are new to programming then we recommend starting with the main users guide which contains details of how to "program" using simple text. However, for the experienced programmer who is used to programming in a high level language (such as C, C++ or BASIC), a DLL (Dynamic Link Library) has been provided to allow easy integration into your own software. The DLL is designed to be as compatible as possible with the vast majority of common programming languages. It provides a comprehensive set of functions that make using the Automator very easy. It encapsulates and hides all of the complexity of USB protocols and lets the user simply express their control requirements in terms that relate to the devices attached. For example to run a stepper motor you would use the function call

AtmRunStepper(steps, direction)

eg

AtmRunStepper(200, 1) ... runs forward by 200 steps

Behind the scenes this function call would automatically find and open a path to the attached Automator board and then carry out the specified function. The opening and closing of such paths is all very speed efficient and transparent to the user. This leaves the user free to concentrate on automation and not USB housekeeping details. The details of each of the DLL functions and examples of how to use them follows below.

2. DLL Reference

The following is a list of all the DLL functions available with a description of what they do and how to use them.

NOTE: For simplicity all function parameters and return values have been made standard integers (32bit)

AtmRunStepper

Run the stepper motor a number of steps forward or reverse.

Format: **AtmRunStepper (steps , direction)**

steps: The number of steps to move
This must be in the range of 1 to 5000000

direction: should be 0 for reverse, or 1 for forward

Example: **AtmRunStepper(500, 0)**
runs stepper 500 steps in reverse

Return Value: A return value of 1 indicates success, 0 for error. Only likely error is board not found(not connected)

Note: The step interval (i.e. the time between steps) is set separately using the AtmSetStepInterval() function. The AtmRunStepper function will always use the current setting for step interval. The power on default is 5ms.

AtmSetStepInterval

Set the step interval to be used when the stepper motor is running.

Format: **AtmSetStepInterval (interval)**

interval: The step interval in milliseconds
This must be in the range of 1 to 30000

Example: **AtmSetStepInterval (25)**
sets the step interval (time between steps) to 25 milliseconds

Note: the step interval can be set at any time, even while the stepper motor is running and will be used immediately.

AtmHaltStepper

Halt the stepper motor immediately

Format: **AtmHaltStepper ( )**

Example: **AtmHaltStepper ( )**
stops the stepper motor running no matter how many steps it has left to complete

AtmSetServo

Set the specified servo to the position.

Format: **AtmSetServo (servonum, position)**

servonum: Identifies which servo to be moved
This must be in the range of 1 to 4 and corresponds to the 4 connection points on the board

position: should be in the range of 1 to 254. These figures correspond to extreme left and right (127 is centre)

Example: **AtmSetServo (3, 210)**
set servo number 3 to position 210

Return Value: A return value of 1 indicates success, 0 for error. Only likely error is board not found(not connected)

AtmSetOutput

Set one of the digital outputs on or off.

Format: **AtmSetOutput (outputnum, onoff)**

outputnum: Identifies which output to be set on or off
This must be either 1 or 2 and corresponds to the two digital output connection points on the board

onoff: '1' for on and '0' for off

Example: **AtmSetOutput (2, 1)**
turns on output 2

Return Value: A return value of 1 indicates success, 0 for error. Only likely error is board not found(not connected)

AtmReadInputs

Reads the current state of the digital inputs.

Format: **AtmReadInputs ( )**

Return value: The current binary value of all inputs

Example: **inputs = AtmReadInputs()**
after the function call the integer variable "inputs" will hold the binary value of the inputs (i.e. bit 0 is input 1 and bit 1 is input 2.

Therefore 0 = both OFF
1 = input 1 ON, input 2 OFF
2 = input 1 OFF, input 2 ON
3 = both ON

"ON" can also be regarded as high, +5v or logic '1'
"OFF" is therefore low, 0v or logic '0'

AtmGetStepsRemaining

Finds out how many steps the stepper still has to complete.

Format: **AtmGetStepsRemaining ( )**

Return value: The current number of steps remaining to be completed

Example: **steps_remainings = AtmGetStepsRemaining ()**

after the function call the integer variable "steps_remaining" will hold the number of steps that the stepper motor still has to complete.

AtmReset

Sets the Automator board back to power on defaults. This will halt the stepper motor and change its step interval back to default(5ms). It will turn off all outputs and set all servos to mid position (127)

Format: **AtmReset ( )**

Example: **AtmReset ( )**

restores the board to power on default settings.

Return Value: A return value of 1 indicates success, 0 for error. Only likely error is board not found(not connected)

3. Using the DLL with Basic (eg visual basic)

Apart from using the functions as described above, you only need to remember to declare the functions correctly so that BASIC can find it in the DLL library. For example the AtmRunStepper() function would be declared at the head of your program as follows....

```
Declare Function AtmRunStepper Lib "atm.dll" (ByVal steps As Integer, ByVal direction As Integer) As Integer
```

The DLL "atm.dll" should be available within the working directory of your program where it will be found whenever you run your program.

Using this function is then simply..

```
Dim steps As Integer
Dim direction As Integer

steps=300
direction=1
AtmRunStepper(steps, direction)                 // or even more simply AtmRunStepper(300,1)
```

4. Using the DLL with "C" (eg C++)

With 'C' you first have to create a function pointer to each function in the DLL before you use it in your program. This means creating a variable to hold that pointer and then making it "point to" the function in the DLL. This is shown below for the AtmSetServo() function.

```
#include "at.h"          // this is the header file provided with the DLL holding the function type definitions

Type_AtmSetServo    AtmSetServo;        // creates a pointer of the correct type to point
                                        // to the AtmSetServo function

HINSTANCE   HAtmDll;                    // handle to be used for the dll
```

Within your initialisation function.....

```
// link to the DLL library which should be located in your working directory
HAtmDll = LoadLibrary("atm.dll");

// set the function pointer to point to the actual function in the DLL
AtmSetServo = (Type_AtmSetServo)GetProcAddress( HAtmDll, "AtmSetServo");
```

Using it throughout your program....

```
int servonum, servoposition;            // declared as simple integers

servonum = 2;
servoposition = 127;
AtmSetServo(servonum, servoposition);   // set servo 2 to position 127 (mid range)
```

5. Terms of Use for all Goods Supplied

Definitions

'Supplier' shall mean PC Control Ltd.

'Buyer' shall mean the person, company or any other body that purchases or agrees to purchase Goods.

'Goods' shall mean all goods and services which the Buyer agrees to buy from the Supplier including replacements for defective Goods, hardware, documentation and software products licensed for use by the Buyer.

Use of the Goods in any way by the Buyer constitutes acceptance of these terms and conditions.

Terms and Conditions

1. The Goods are intended to be part of the buyer's own design of apparatus and not a finished product in their own right.
2. The Goods supplied are not to be used in any design where there is a risk, however small, either directly or indirectly, of death or personal injury.
3. The Buyer will be responsible for ensuring the fitness for purpose of the Goods for the Buyer's application.
4. To the extent permitted by law, the Supplier accepts no liability whatsoever or howsoever arising in respect of loss, damage or expense arising from errors in information or advice provided whether or not due to the Supplier's negligence or that of its employees, agents or sub-contractors save for any loss or damage arising from death or personal injury.
5. To the extent permitted by law, the Supplier shall not be liable to the Buyer by reason of any representation (unless fraudulent), or any implied warranty, condition or other term, or any duty at common law, or under the express terms of any Contract with the Buyer, for any indirect, special or unforeseen loss or damage (whether for loss of profit or otherwise), costs, expenses or other claims for compensation whatsoever (whether caused by the negligence of the Supplier, its employees or agents or otherwise) which arise out of or in connection with the supply of the Goods or their use or resale by the Buyer.
6. The entire liability of the Supplier under or in connection with the Contract with the Buyer shall not exceed the price of the Goods except as expressly provided in these terms and conditions.
7. Before proceeding with the installation and use of this software, carefully read the following terms and conditions of this license agreement and limited warranty ("the agreement").By installing or using this software you indicate your acceptance of this agreement. If you do not accept or agree with these terms, you may not install or use this software. This software, including documentation is owned by PC Control Ltd. This agreement does not provide you with title or ownership of the software, but only a right of limited use as outlined in this license agreement. When purchased, PC Control ltd. will grant you a non-exclusive, royalty free license to use the software on the single computer on which it is installed. The license is also restricted to the specific hard drive on that computer that is present during installation. The software is supplied solely for use in conjunction with our range of boards. Without the written consent of PC Control ltd., you may not rent or lease the software, or sell any portion of the software. This software is provided "as is" without warranty of any kind either expressed or implied, including ,but not limited to, the implied warranties of merchantability and fitness for a particular purpose. PC Control Ltd. do not warrant that the functions contained in the software will meet your requirements or that the operation of the software will be uninterrupted or error free. In no event shall PC Control Ltd or any other party who may have distributed the software be liable for damages, including any general, special incidental, or consequential damages arising from the use or inability to use the software, including, but not limited to, loss of data or losses sustained by you or third parties.
8. These terms are an important part of the full terms and conditions of business as published on the website at www.pc-control.co.uk/general-terms.htm which also apply.